Lüke, Laura; Heßler, Katrin; Irnich, Stefan

The single picker routing problem with scattered storage: modeling and evaluation of routing and storage policies

OR Spectrum

Suggested Citation: Lüke, Laura; Heßler, Katrin; Irnich, Stefan (2024) : The single picker routing problem with scattered storage: modeling and evaluation of routing and storage policies, OR Spectrum, ISSN 1436-6304, Springer Berlin Heidelberg, Berlin/Heidelberg, Vol. 46, Iss. 3, pp. 909-951,
https://doi.org/10.1007/s00291-024-00760-4

OR Spectrum (2024) 46:909–951
https://doi.org/10.1007/s00291-024-00760-4

ORIGINAL ARTICLE

The single picker routing problem with scattered storage: modeling and evaluation of routing and storage policies

Laura Lüke[1] · **Katrin Heßler**[2] · **Stefan Irnich**[1]

Received: 22 June 2023 / Accepted: 26 February 2024 / Published online: 19 May 2024

Abstract

Despite ongoing automation efforts, most warehouses are still manually operated using a person-to-parts collection strategy. This process of collecting items of customer orders from different storage locations accounts for the majority of the operating costs of the warehouse. Hence, optimizing picker routes is an important instrument to reduce labor costs. We examine the scattered-storage variant of the single picker routing problem in a one-block parallel-aisle warehouse. With scattered storage, an article can be stored at several storage locations within the warehouse, whereas with classic storage, each article has a unique storage location. We use our recently published network-flow model with covering constraints that is based on an extension of the state space of the dynamic-programming formulation by Ratliff and Rosenthal. With modifications in the state graph, this model serves for both exact and all established heuristic routing methods for picker routing. The latter include traversal, return, largest gap, midpoint, and composite. We show that these routing policies can also be implemented through adaptations in the state space. Extensive computational studies highlight a comparison of the different routing and storage policies (in particular class-based storage policies) in the scattered storage context. Analyses demonstrate which combinations of policies are advantageous for the given warehouse layout. For class-based storage policies, we emphasize how the scattering of articles of different classes should be performed: scattering of C-articles is advantageous with reductions of up to 25%. In contrast, when articles are uniformly distributed, A-articles should be scattered.

Keywords Routing · Warehousing · Picker routing · Scattered storage · Storage policy

Extended author information available on the last page of the article

1 Introduction

In this work, we consider the *single picker routing problem* (SPRP), which is one of the fundamental problems in warehouse operations for manual (non-automated) warehouses. For a given warehouse layout and a list of articles with a storage location (=pick position) for each of the articles, the SPRP consists of deciding how a picker travels through the warehouse in order to collect the articles (picker-to-parts) in a minimum-length tour. It is estimated that more than 80% of all order-picking systems in Western Europe are low-level picker-to-parts picking systems (de Koster et al. 2007). A study by Behnisch et al. (2017) highlights that 60% of companies in Germany that have small-sized warehouses (<10,000 square meters) use paper-based pick lists so that the pickers are routed by heuristics.

We study the extension of the SPRP to warehouses *with scattered storage* (abbreviated as SPRP-SS in the following) under the combination of different *routing policies* and *storage policies*. The three emphasised terms need to be further elaborated: First, a warehouse operates as a scattered storage warehouse or mixed-shelves warehouse if one or several articles are pickable from more than one pick position. Scattered storage is predominant in modern e-commerce warehouses of companies like Amazon or Zalando (Weidinger 2018; Weidinger and Boysen 2018; Boysen et al. 2019; Weidinger et al. 2019; Goeke and Schneider 2021; Bock and Boysen 2023; Khan et al. 2024). The main advantage of a scattered storage strategy is "that items of demanded SKUs are found close by irrespective of the position within the warehouse [so that] the distance to be covered for order picking is reduced this way" (Weidinger 2018, p. 139). Boysen et al (2019, p. 399) point out that warehouses with scattered storage often have multiple I/O points where completed orders are handed over to the central conveyor system. As a result, the length of picker tours further decreases and tight delivery schedules can better be met. Another advantage is that scattered storage comes without excessive automation. In particular an adaption to varying workloads is easily possible by using less or more pickers. Our own study (see Sect. 6.3) reveals that it is of high importance to determine which article should be placed where and how often at different positions in the warehouse. We show that different strategies can easily lead to picker tours that, on average, differ in length by more than 30%.

The term scattered storage should not be confused with *shared storage*: while shared storage allows to allocate a storage location consecutively to different articles in general (Cormier and Gunn 1992), scattered storage even allows a single load unit to be broken up and distributed to different storage locations (Weidinger and Boysen 2018).

Second, even for the most simple warehouse layout, i.e., a single-block warehouse with parallel aisles, exact (=minimum length) picker tours can be complicated, counter-intuitive, and difficult to memorize, see, e.g., (Petersen 1997, p. 1102). Therefore, the application of routing heuristics can be justified: the pickers can only perform tours defined by some simple rules. We consider the standard rule-based routing policies such as `traversal` (a.k.a. S-shape), `midpoint`, `largest gap` (Hall 1993), `return`, and `composite` (Petersen 1997).

Third, articles are either stored at a dedicated pick position, chaotically at random storage locations, or the warehouse is grouped into zones (de Koster et al. 2007). In the latter case, the articles are often assigned to these zones on the basis of their turnover rate (Petersen and Schmenner 1999). Different *class-based storage policies* can use, e.g., an ABC-classification of articles to determine which article is allocated to which storage locations in the warehouse.

1.1 Definition of the SPRP-SS

For a formal definition of the SPRP-SS, we assume that S denotes the set of different articles that a picker must collect in a single picker tour. Moreover, $q_s \in \mathbb{N}$ units (=items) of this article $s \in S$ need to be collected from one or several given positions P_s, where $b_{sp} \in \mathbb{N}$ units of article s are available at position $p \in P_s$. The special case of $q_s = 1$ for all articles $s \in S$ is known as the *unit-demand* case, for which also $b_{sp} = 1$ can be assumed w.l.o.g. The unit-demand case also describes the situation that sufficiently many items of all articles are available so that any demand can be collected from a single position for each article s. Otherwise, if $q_s > 1$ for some $s \in S$, the SPRP-SS instance describes the *general-demand* case. Here, it may be necessary or advantageous to collect the same article from different positions. In the standard variants of the SPRP-SS, pickers start and end their tour at the same point in the warehouse often denoted as the I/O point 0 (=depot).

Let $P_0 = \{0\}$ and $P = P_0 \cup \bigcup_{s \in S} P_s$, where the latter set comprises the set of all positions relevant for the SPRP-SS. A tour is feasible for the given SPRP-SS instance, i.e., it allows the collection of the entire demand, if the tour visits a subset $P' \subseteq P$ such that $0 \in P$ and $\sum_{p \in P' \cap P_s} b_{sp} \geq q_s$ for all $s \in S$. Figure 1 shows a feasible tour for an SPRP-SS instance with general demand. Let the $D = (d_{pq})$ be the (symmetric) distance between two positions $p \in P$ and $q \in P$. A tour is optimal, if it is feasible and minimizes the distance traveled by the picker using the distance matrix D. In the unit-demand case, the SPRP-SS instance can be modeled and

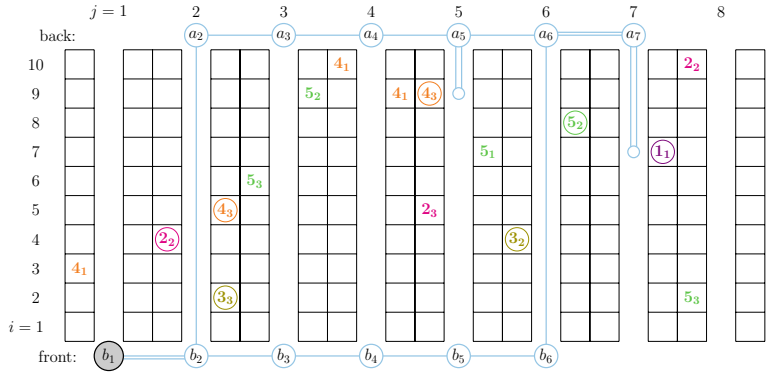


Fig. 1 A feasible tour for a general-demand SPRP-SS instance with demand $q_1 = q_2 = q_5 = 1$ and $q_3 = q_4 = 5$. The pick positions of the five articles $S = \{1, 2, 3, 4, 5\}$ are indicated with the article s and an index. The index shows the available number b_{sp} of items of the article s at the pick position p. Pick positions from where the picker collects items are encircled

solved as a special case of the *generalized traveling salesman problem* (GTSP, Gutin and Punnen 2002): The set of cities is given by P, the distances by $D = (d_{pq})$ and the clusters by P_0 and P_s for $s \in S$.

The standard SPRP (non-scattered) is the special case that all articles are available at a unique position, i.e., $|P_s| = 1$ for all $s \in S$. In contrast to the SPRP-SS, there is no selection of positions required, but every tour that covers all given positions is feasible. The SPRP seeks a minimum-length picker tour given the distances and the pick positions from where articles must be collected. Hence, it is a special case of the *(Steiner) traveling salesman problem* (TSP, Cornuéjols et al. 1985; Roodbergen 2001). While the TSP is generally NP-hard, the modeling and solution approaches for picker routing heavily rely on the fact that typical warehouses have a well-defined structure that can be exploited. We discuss this in the following literature review.

1.2 Literature review

Literature on warehouse operations is extensive. For the sake of brevity, we restrict ourselves to the same three central topics already discussed in the introduction, i.e., picker routing, scattered storage, and class-based storage policies. Intentionally, we do not discuss integrated problems in warehouse operations management where picker routing constitutes a well-defined subproblem such as in order batching and batch scheduling (van Gils et al. 2019) or warehouse layout and storage design problems (Henn et al. 2013).

For a single-block parallel-aisle warehouse, the seminal work of Ratliff and Rosenthal (1983) has shown that the SPRP is a well-solvable special case of the TSP. Recently, Heßler and Irnich (2022b) stated the complexity of the SPRP more precisely as linear in the number of aisles and the number of pick positions. The generalization of this dynamic-programming (DP) algorithm to the case of a two-block parallel-aisle warehouse was presented by Roodbergen and de Koster (2001b). For an arbitrary number of blocks, Pansart et al. (2018) presented a DP as well as a MIP-based approach. Despite the low computational complexity of computing optimal picker tours with DP algorithms, the application of heuristic policies is well justified in settings where pickers prefer to perform tours defined by some simple rules. Accordingly, *heuristic routing policies* are rule-based heuristics (see Sect. 2). In the following, we also consider the use of minimum-length tours and denote the routing policy by `exact`. Both exact and heuristic techniques have been extended into many different directions, e.g., to various warehouse layouts (Roodbergen and de Koster 2021a,b; Öztürkoğlu et al. 2012; Çelk and Süral 2014), non-identical start and endpoints (Masae et al. 2020; Löffler et al. 2022a), and multiple end depots (de Koster and van der Poort 1998; Goeke and Schneider 2021). The most recent survey on picker routing is the one of Masae et al. (2020).

In Table 1, we provide an overview of SPRP and SPRP-SS variants that have been discussed in the literature. Variants address different problem dimensions such as the layout of the warehouse, the number of I/O points (=depots), and the routing policies.

Table 1 Problem dimensions of the SPRP and SPRP-SS

Dimension	Attribute	References	DP known
Layout	Single-block	Ratliff and Rosenthal (1983)	Yes
	Two-block	Roodbergen and de Koster (2001b)	Yes
	Multi-block	Pansart et al. (2018)	Yes
	Flying-V	Gue and Meller (2009)	Yes
	Butterfly	Öztürkoğlu et al. (2012)	No
	Fishbone	Çelk and Süral (2014)	Yes
	Chevron	Masae et al. (2019)	Yes
	Discrete cross-aisles	Öztürkoğlu and Hoser (2019)	Yes
	Leaf	Masae et al. (2021)	Yes
I/O point(s)	One unique point (= depot)	Ratliff and Rosenthal (1983)	Yes
	Multiple drop-off points	de Koster and van der Poort (1998)	Yes
	Non-identical start and drop-off point	Masae et al. (2020); Löffler et al. (2022b)	Yes
Routing policy	Exact	Ratliff and Rosenthal (1983)	Yes
	`traversal`, `midpoint`, `largest gap`	Hall (1993)	Yes
	`return`	Petersen (1997)	Yes
	`composite`	Petersen (1997)	No/yes[¶]
	`combined`	Roodbergen and de Koster (2001a)	Yes[‡]
Demand[†]	Unit demand	Daniels et al. (1998)	n.a
	General demand	Daniels et al. (1998)	n.a

Similar to Table 1 in (Heßler and Irnich 2024)

[¶]: Different definitions of the policy were given by Petersen (1997), Scholz and Wäscher (2017b), and *this paper*, DP known only for the last definition

[‡]: Different type of DP. [†]: Only relevant for the SPRP-SS

Scattered storage was first addressed by Daniels et al. (1998), who also pointed out that several items of the same article can or must be retrieved from different pick positions whenever the demanded number of items is greater than one (*general-demand case*). For the general-demand case, the authors proposed a TSP-type of formulation with demand fulfillment constraints as well as a solution approach using a tabu search metaheuristic for the selection of pick positions. For the *unit-demand* case, Daniels et al note that the resulting SPRP-SS is a special case of *generalized traveling salesman problem* (GTSP, Gutin and Punnen 2002). Although Gu et al. (2007) already stated the great research potential of the SPRP-SS, it received only little attention up to the middle of the last decade. Weidinger (2018) coined the term *mixed-shelves storage* as a synonym for scattered storage, emphasising that it is particularly advantageous in large warehouses with many different articles but rather small, time-critical orders. Weidinger showed that, for a single-block parallel-aisle warehouse, the determination of a minimum-length picker tour is NP-hard. Moreover, he compared a decomposition procedure (select the pick position by different priority rules and use the algorithm of Ratliff and Rosenthal to determine a picker

tour) with the MIP-approach of Daniels et al. (1998) complemented with MTZ-based subtour-elimination constraints (Miller et al. 1960). The effective model of Goeke and Schneider (2021) is tailored to the single-block parallel-aisle warehouse layout. Independently, a rather involved MIP-based approach has been presented by Su et al. (2023) for multi-block parallel-aisle warehouses. Unfortunately, this approach has not been compared to any GTSP-based model or the approach of Pansart et al.. The currently fastest exact SPRP-SS algorithm is the one of Heßler and Irnich (2024) which is explained and exploited in Sects. 2 and 3. Their approach assumes that the warehouse layout and I/O point configuration are such that a DP formulation for the non-scattered SPRP is known and can be extended to capture the options arising from scattered storage. The latter is true as can be seen from the last column of Table 1. A rather involved MIP model (solved with a MIP solver) has been presented by Su et al. (2023) for the SPRP-SS defined for multi-block parallel-aisle warehouses and the unit-demand case. Unfortunately, this approach has not been compared to any GTSP-based solution approach. Overall, all exact solution approaches for the SPRP-SS are MIP-based. Complementing exact approaches, Weidinger et al. (2019) presented a heuristic solution approach for SPRP-SS with multiple depots.

Class-based storage was first introduced by Hausman et al. (1976). Different criteria that are often used for the assignment of articles to classes are the *turnover rate*, the *required space*, and the *cube-per-order index* (COI) (Heskett 1963) which is the ratio of required space to turnover rate (Gu et al. 2007). Gibson and Sharp (1992) showed that class-based storage leads to significantly shorter picker tours than random storage. A special case of class-based storage is *full-turnover storage* where all articles are ranked according to their turnover rate. The articles are then assigned to the pick positions in such a way that articles with the highest turnover rate are located closest to the depot and the articles with the smallest turnover rate are most distant to the depot. Thus, in this case, each article is assigned to a different class (de Koster et al. 2007). Full-turnover storage leads to shorter picker tours than class-based storage but is more complex to implement and more detailed data is needed (Petersen et al. 2004). Hence, Petersen et al. (2004) recommend class-based storage with two to four classes. Park and Webster (1989) consider three-dimensional storage systems (vertically and horizontally organized storage positions) with specific handling equipment and suggest a so-called 'cubic-in-time' rule for designing a two-class storage warehouse minimizing traveling time.

1.3 Contributions

This is the first (exact) approach for picker routing that combines heuristic routing policies with scattered storage. We formally introduce variants of the SPRP-SS in which picker routes must obey the rules of the heuristic routing policies `traversal`, `return`, `largest gap`, `midpoint`, and `composite`, respectively. The contributions of this work are:

- We prove that all variants of the SPRP-SS with the above heuristic routing policies and scattered storage remain NP-hard.
- Adapting the approach of Heßler and Irnich (2024) for the routing policy `exact` to heuristic routing policies, we provide a first `exact` approach for the SPRP-SS with the routing policies `traversal`, `return`, `largest gap`, `midpoint`, and `composite`. Our solution approach is based on an incomplete DP formulation finally solved as an *integer program* (IP) with the help of a *mixed-integer programming* (MIP) solver. In this context, 'incomplete' means that the DP formulation does not ensure demand covering for articles available in more than a single aisle.
- We analyze combinations of routing and storage policies. In this sense, our work extends classical results of Petersen and Schmenner (1999) to warehouses with scattered storage. We find that larger cost savings are possible with scattered storage by choosing a suitable combination of the routing and storage policy compared to *classical warehouses*, i.e., those without scattered storage. The combination of `largest gap` and *within-aisle* outperforms other combinations in most cases.
- For class-based storage policies, we investigate how the scattering of articles of different classes should be performed. For *uniformly distributed* articles, scattering of A-articles is most beneficial as suggested by Weidinger (2018). We also obtain an unexpected result: scattering of C-articles performs best for the standard class-based storage policies.

1.4 Structure

The remainder of this paper is structured as follows. In Sect. 2, we briefly review the DP formulation of Ratliff and Rosenthal (1983) and detail the necessary modifications on the DP state space to enforce picker routes that obey the rules of the respective heuristic routing policy. In Sect. 3, we elaborate which extensions of the state space are required to correctly model the options arising from scattered storage. Together, these results allow us to specify a unified integer programming formulation. The NP-hardness of the SPRP-SS even for simple heuristic routing policies is proven in Sect. 4. Section 5 formally introduces the class-based storage policies. The comprehensive computational study is presented in Sect. 6. The work closes with conclusions and an outlook in Sect. 7.

2 Solving the SPRP for different routing policies

While the non-scattered standard SPRP can be solved efficiently, the SPRP-SS is provenly NP-hard (Weidinger 2018). Different solution approaches for optimal routing (hereafter denoted as `exact` routing policy) have been presented in the literature (see Sect. 1.2). To date, we are not aware of any exact algorithm for the SPRP-SS and heuristic routing policies. In the following, we rely on an exact algorithm first presented in (Heßler and Irnich 2022a) (this technical report will remain unpublished) and Heßler

and Irnich (2024). Their approach assumes that the warehouse layout is such that a DP formulation for the non-scattered SPRP is known and can be extended to capture the options arising from scattered storage. The focus of the companion paper is on algorithmic performance, as it compares the solution times of the available solution approaches for SPRP-SS applicable to one-block and two-block parallel-aisles warehouses and the `exact` routing policy. We would like to stress that the paper at hand has a completely different focus, i.e., different heuristic routing policies evaluated in combination with different storage policies. Particularly, computation times are irrelevant because they are in the range of milliseconds (never exceeding one second, see Sect. 6.2). To make the paper at hand self-contained, we present the ideas coined in (Heßler and Irnich 2022a) in the following paragraph.

Heßler and Irnich (2022a, 2024) have shown that, for the routing policy `exact` in a single-block parallel-aisle warehouse, the SPRP-SS can be solved as a shortest-path problem with additional covering constraints. The underlying network is an extension of the state space of Ratliff and Rosenthal's DP. We adapt the approach to heuristic routing policies. Section 2.1 introduces the notation and original state space, while Sect. 3 explains the necessary extension in the form of additional actions per aisle. In the extended state space, every picker tour is still a path, and vice versa. Exploiting the equivalence of DP and linear programming, a network-flow formulation for an origin–destination shortest path problem can be set up. The requirement to make consistent selections of pick positions can be modeled with additional covering constraints. The resulting IP formulation is also presented in Sect. 3.

In this work, we follow the same solution approach for solving the SPRP-SS but now for the different heuristic routing policies. The new idea presented here is that the picker tour defined by a heuristic routing policy is a shortest path in a DP state space that is adapted according to the routing policy. Accordingly, we construct state spaces for the heuristic routing policies (Sects. 2.2–2.6). We summarize the results of this longer DP-related part in Sect. 2.7.

2.1 Routing policy exact

Let $J = \{1, 2, \dots, m\}$ denote the set of aisles (numbered from left to right). The DP formulation of Ratliff and Rosenthal (1983) constructs a distance-minimal picker tour by alternately deciding how the picker moves through an aisle $j \in J$ and through the next cross-aisles connecting aisle j with aisle $j + 1$. Accordingly, the stages of the DP can be denoted by

$$1^-, 1^+, 2^-, 2^+, \dots, m^-, m^+, (m+1)^-,$$

where j^- (j^+) represents the situation before (after) the aisle $j \in J$ is traversed. Likewise, j^+ and $(j+1)^-$ represent the situations before and after the cross-aisle action connecting aisle j with $j + 1$ has been performed, respectively. The stage $(m + 1)^-$ is added to have a unique final state (see below).

For a distance-minimal picker tour, within an aisle, only the actions

$$E^{aisle} = \{\texttt{1pass}, \texttt{2pass}, \texttt{top}, \texttt{bottom}, \texttt{gap}, \texttt{void}\},$$

are possible, where 1pass, 2pass, stands for a single (double) traversal through the aisle (in either direction), top (bottom) for a traversal from the back (front) cross-aisle to the lowest (highest) relevant pick position and back, gap for double-sided traversal from front and back cross-aisle leaving a maximum length gap in the middle, and void for no traversal of the aisle, which is only possible in aisles from where nothing needs to be picked.

Possible cross-aisle actions are

$$E^{cross} = \{00, 11, 20, 02, 22\},$$

where the first (second) digit gives the number of traversals of the back (front) cross-aisle.

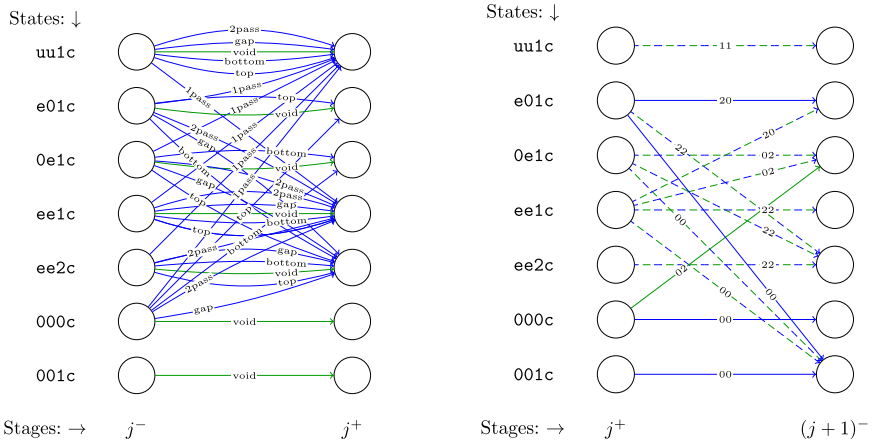
Ratliff and Rosenthal introduced so-called *partial tour subgraphs* (PTSs) that represent the part of the picker tour containing only actions between aisle 1 and aisle j. A PTS results from introducing vertices a_j and b_j at the crossing points of aisle j and the back and front cross-aisle, respectively (see Fig. 3). Every PTS can be characterized by the following set of states:

$$\mathscr{S} = \{\texttt{uu1c}, \texttt{e01c}, \texttt{0e1c}, \texttt{ee1c}, \texttt{ee2c}, \texttt{000c}, \texttt{001c}\}.$$

The first two symbols describe the parity of the right-most vertices a_j and b_j with u, e, and 0 for uneven (=odd), even (with positive degree), and degree 0. The last two symbols describe the number of connected components of the PTS, i.e., 1c, 2c, for one (two) connected components, and 0c for an (empty) PTS without edges.

Figure 2 visualizes the states and actions for the routing policy exact. The state space of Ratliff and Rosenthal's DP can be constructed by joining the building block depicted in Fig. 2a for $j = 1$ with the building block depicted in Fig. 2b for $j = 1$, the latter one with the building block depicted in Fig. 2a for $j = 2$, etc. The initial state is $o = \texttt{000c}$ at stage 1^-, and the final state is $d = \texttt{001c}$ at stage $(m + 1)^-$. The resulting state space is depicted in Fig. 3.

In contrast to Ratliff and Rosenthal, we do not model the depot 0, which is located at the front or back of the depot aisle j_0, as a pick position. Instead, the possible actions in cross-aisles can be restricted so that the depot is surely connected with the constructed tour, see the caption of Fig. 2b. For example, when being in state e01c at stage j^+, the actions 00 and 20 would make b_j disconnected, which is forbidden if j is the aisle j_0 of the depot 0 (in this case b_j represents the depot). Note that these restrictions are based on the assumption that the depot 0 is located at a front cross-aisle. In the case that the depot is at the back cross-aisle, the roles of back and front must be swapped. For simplicity, to not display symmetric cases, we assume in the following figures that the depot 0 is located at the front cross-aisle. Note also that our modeling approach is valid for any *depot aisle* $j_0 \in J$, not only if $j_0 = 1$, i.e., the depot is at the left end. Figure 3 illustrates an example SPRP instance where the depot is located in front of aisle $j_0 = 3$. Figure 3a pictures the warehouse with the depot, the pick positions of the different (numbered) articles, and the optimal picker tour. The corresponding state space is shown in Fig. 3b, where the optimal control describing the optimal picker tour is marked in red. It is the shortest path from the initial state o to the final state d. Looking at the optimal picker tour shown



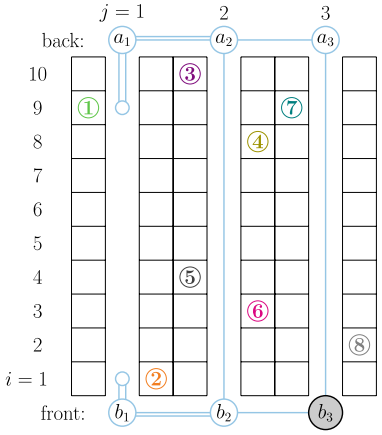
(a) Actions within an aisle j. Note that green (blue) arcs are only valid in (non-)empty aisles.

(b) Actions of the cross-aisles. Note that green (blue) arcs are only valid in (non-)depot cross-aisles, i.e., if $j = j_0$ ($j \neq j_0$). Dashed arcs are valid in all cross-aisles.

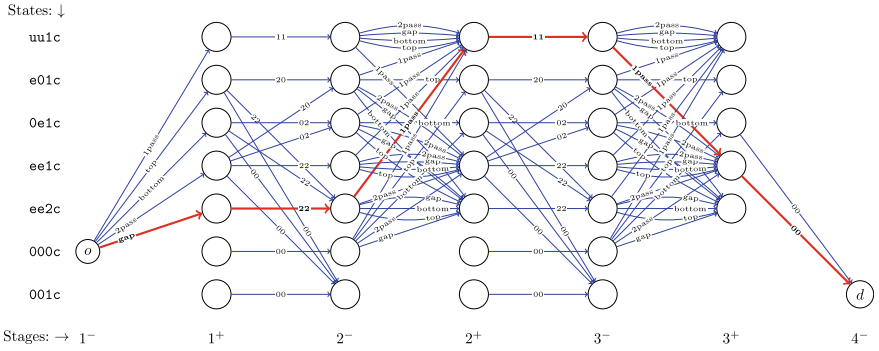
Fig. 2 Possible actions between states for the routing policy `exact`

in Fig. 3a, we briefly discuss the PTS of each stage: At stage 1^-, the PTS does not include any parts of the picker tour. At stage 1^+, aisle $j = 1$ is traversed using the action `gap`. Thus, both vertices a_1 and b_1 have an even parity with a degree of two, leading to a PTS with state `ee2c`, as the PTS consists of two components. The subsequent PTS at stage 2^- does not change its state because passing the cross-aisle sections between a_1 and a_2 and b_1 and b_2 twice still results in two components with an even degree. At stage 2^+, aisle $j = 2$ is traversed using the action `1pass` changing the degree of a_2 and b_2 to three. As the PTS becomes connected, the corresponding state is `uu1c`. With the cross-aisle action `11`, a_3 and b_3 have degree one so that the state of stage 3^- is `uu1c`. Following the same procedure with actions `1pass` and `00`, the remaining PTSs of this instance have the states `ee1c` and `001c` for stages 3^+ and 4^-, respectively.

To complete the definition of the DP, one must associate a cost to all actions, i.e., the arcs of the state graph: the cost is defined by the distance traveled (or time consumed) by the picker when performing the associated action. Recently, Heßler and Irnich (2022b) have shown that computing these costs can be done with linear effort measured in the number m of aisles and number n of pick positions. Recall that any o-d-path in the state space represents a feasible picker tour. As a result, the length of the picker tour is exactly the cost of the path. Therefore, solving the SPRP can be interpreted as solving an origin–destination shortest-path problem. Note that the overall complexity is also linear because there is only a linear number of states and actions and the state graph is acyclic.



(a) Instance and optimal picker tour.



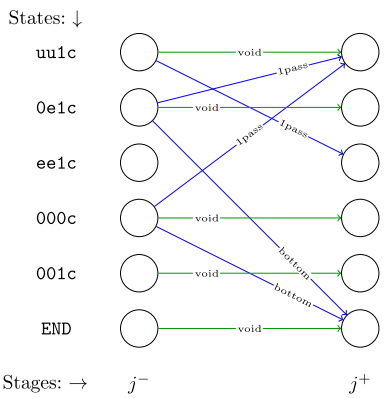
(b) State space and shortest o-d-path corresponding to the optimal picker tour (marked in red/thick).

Fig. 3 SPRP instance with the optimal picker tour and the corresponding state space

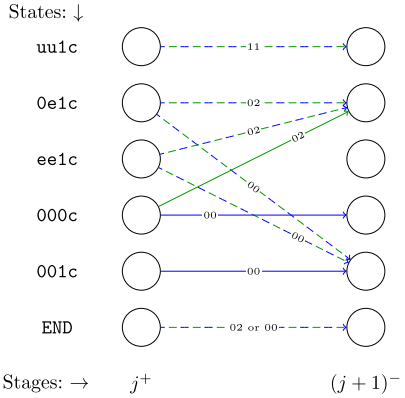
2.2 Routing policy traversal

The rules of `traversal` specify that an aisle is completely traversed when it is visited (Hall 1993). This means that if an aisle is entered from the back (front), it needs to be left at the front (back) cross-aisle. However, there is an exception regarding the rightmost aisle entered by the picker tour. In case an odd number of aisles is traversed in the course of the picker tour, this last aisle is not traversed completely, but entered and left from the same end. Thus, after collecting the articles, the picker is on the same side of the warehouse as the depot and can move back there without crossing an additional, unnecessary aisle. In particular, if the order only contains articles that are stored in one single aisle, then this aisle is not traversed completely.

For the remainder of this section, we assume the depot 0 to be located within the front cross-aisle at the level of the leftmost aisle, i.e., $j_0 = 1$. Then, possible



(a) Actions within an aisle j. Note that `bottom` is only valid in the right-most non-empty aisle and green (blue) arcs are only valid in (non-)empty aisles.



(b) Actions of the cross-aisles. Note that green (blue) arcs are only valid in (non-)depot cross-aisles, i.e., if $j = j_0$ ($j \neq j_0$). Dashed arcs are valid in all cross-aisles. The action from state `END` to `END` is `02` (`00`) for $j + 1 \leq j_0$ ($j + 1 > j_0$).

Fig. 4 Possible actions between states for the routing policy `traversal`

actions within an aisle and possible cross-aisle actions for the routing policy `traversal` are

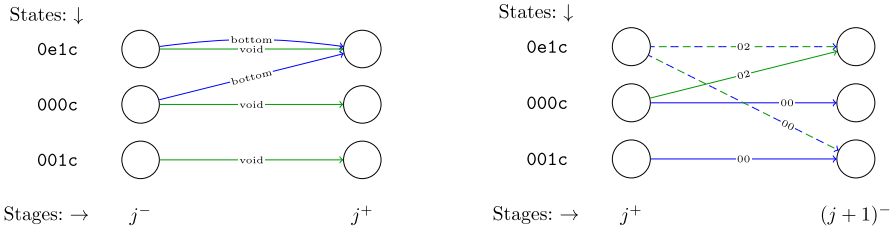
$$E^{aisle} = \{\texttt{1pass}, \texttt{bottom}, \texttt{void}\} \qquad \text{and} \qquad E^{cross} = \{\texttt{00}, \texttt{11}, \texttt{02}\}.$$

When the depot is located in the back cross-aisle, actions `top` and `20` replace `bottom` and `02`.

Due to the smaller sets of actions (compared to `exact`), the set of states for `traversal` has fewer elements. State `ee2c` represents a PTS consisting of two components and is impossible here (`ee2c` cannot be reached by any heuristic routing policy). Additionally, state `e01c` is omitted from the state set if the depot is located at the front cross-aisle, since with the allowed actions of `traversal`, the state `e01c` cannot be reached. Instead, `0e1c` is omitted if the depot is sited at the back cross-aisle. Apart from the two excluded states, there is also a supplementary state denoted by `END`. This state can only be reached in combination with the action `bottom` (or `top`, if the depot is located in the back cross-aisle) and is therefore responsible to enforce the above deviating rules for the last aisle entered by the picker tour. The resulting set of states is

$$\mathcal{S} = \{\texttt{uu1c}, \texttt{0e1c}, \texttt{ee1c}, \texttt{000c}, \texttt{001c}, \texttt{END}\}.$$

Figure 4 depicts which state-action combinations are possible. Note that some combinations only exist in (non-)empty aisles or (non-)depot cross-aisles. In addition,



(a) Actions within an aisle j. Note that green (blue) arcs are only valid in (non-)empty aisles.

(b) Actions of the cross-aisles. Note that green (blue) arcs are only valid in (non-)depot cross-aisles, i.e., if $j = j_0$ ($j \neq j_0$). Dashed arcs are valid in all cross-aisles.

Fig. 5 Possible actions between states for the routing policy `return`

the action of the cross-aisles between two END-states varies depending on the location j_0 of the depot 0. If aisle j is left to the depot aisle j_0 (or identical to it), then the action is 02: The picker has already collected all demanded items at this point, but still has to move to the depot. In case j is right to j_0, the action is 00, and the picker does not need to move further but returns directly to the depot.

2.3 Routing policy return

The opposite routing policy to `traversal` is `return` (Petersen 1997). Instead of traversing each relevant aisle completely, for `return` each relevant aisle is entered and left from the same end. The change of direction in the visited aisle is performed as soon as all requested articles in this aisle are collected. As the picker starts and ends the tour at the depot, the location of the depot determines the cross-aisle the picker passes through and, consequently, the end from which aisles are entered. This rather simple routing policy requires only two different actions each, within the aisles and for the cross-aisles. Not even for the last aisle of the picker tour is an additional action needed. Given the depot is located in the front cross-aisle, the two sets of actions are

$$E^{aisle} = \{\texttt{bottom}, \texttt{void}\} \qquad \text{and} \qquad E^{cross} = \{00, 02\}$$

(if the depot is located in the back cross-aisle, instead of `bottom` the action `top` is used, and 02 is exchanged by 20). The limited number of actions imposes three necessary states which are

$$\mathscr{S} = \{\texttt{0e1c}, \texttt{000c}, \texttt{001c}\}.$$

(if the depot is placed in the back cross-aisle, `0e1c` is exchanged by `e01c`).

Figure 5 confirms that the small number of possible actions and states also leads to few possible combinations in the state space.

2.4 Routing policy largest gap

The routing policy `largest gap` has rather complex rules (Hall 1993): An aisle is entered (and left again) from both ends. The two turning points within the aisle are chosen in such a way that the longest possible part of the aisle is not traversed because there are no relevant articles. This part of the aisle is called `largest gap`. When applying `largest gap`, the picker moves through both cross-aisles one after the other and enters relevant aisles in between from the end that borders the cross-aisle. To reach the opposite cross-aisle, the picker traverses the first and last aisle with relevant pick positions completely. For the implementation of these routing rules, the actions

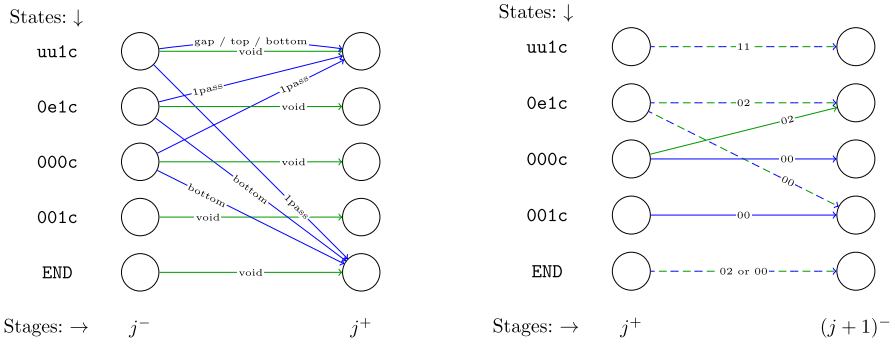
$$E^{aisle} = \{\texttt{1pass}, \texttt{top}, \texttt{bottom}, \texttt{gap}, \texttt{void}\} \qquad \text{and} \qquad E^{cross} = \{\texttt{00}, \texttt{11}, \texttt{02}\}$$

are required. Here, action `1pass` is only used for the first and last aisle. For the aisles in between, the use of `gap` is intended. If only one demanded article is stored in an aisle or if the largest gap of an aisle is connected to one of the two cross-aisles, then this aisle is only entered from one end and the action `top` or `bottom` is used instead of `gap`. Concerning the cross-aisle actions, `02` is replaced by `20`, if the depot is located in the back cross-aisle. The set of possible states necessary for `largest gap` is

$$\mathscr{S} = \{\texttt{uu1c}, \texttt{0e1c}, \texttt{000c}, \texttt{001c}, \texttt{END}\}.$$

Since the first and last aisle are traversed differently than all the other aisles, this must be taken into account when defining the set of possible states. Thus, the additional state `END` is needed to enable the action `1pass` within the last aisle. For the complete traversal of the first aisle of the tour, no extra state is needed. As the initial state `000c` is only used for this aisle (and for empty aisles left of it), only the actions `1pass` and `void` are allowed at this state (see Fig. 6a).

In case all articles of the pick list are stored in one single aisle, the picker moves through the cross-aisle of the depot until the corresponding aisle is reached (in general, in either direction from left to right or right to left). In Fig. 6a, this case can be observed in the states `0e1c` and `000c` with action `bottom` assuming the depot is located at the front cross-aisle. This aisle is then served using the action `bottom` (otherwise with action `top`). Another noticeable state is `uu1c` for which it depends on the number and the distribution of the articles which action `gap`, `top` or `bottom` is possible (in a middle aisle of the tour). Looking at Fig. 6b, the possible action between two `END`-states is again dependent on the depot location.



(a) Actions within an aisle j. Note that green (blue) arcs are only valid in (non-)empty aisles. Moreover, `1pass` and `bottom` with head `END` are only valid in the right-most non-empty aisle. Between two `uu1c` states is either `gap` or `top` or `bottom` allowed.

(b) Actions of the cross-aisles. Note that green (blue) arcs are only valid in (non-)depot cross-aisles, i.e., if $j = j_0$ ($j \neq j_0$). Dashed arcs are valid in all cross-aisles. The action from state `END` to `END` is `02` (`00`) for $j+1 \leq j_0$ ($j+1 > j_0$).

Fig. 6 Possible actions between states for the routing policy `largest gap`

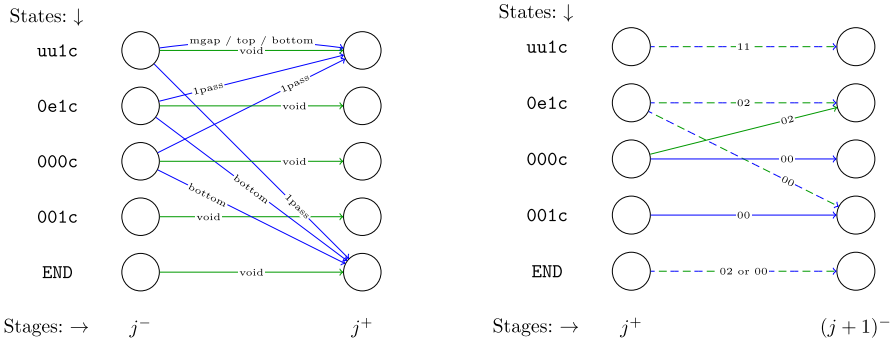
2.5 Routing policy midpoint

The routing policy `midpoint` (Hall 1993) has many similarities to the just-described policy `largest gap`. The picker moves through the warehouse following almost all principles of `largest gap`. The only difference is the criterion for defining the turning points in aisles that are visited from both ends. While for `largest gap` the picker leaves out the maximum length part, the rules for `midpoint` require the picker to not cross the middle of an aisle. This means if the picker enters an aisle from the back (front) cross-aisle, he or she is only allowed to collect articles closer to the back (front) cross-aisle than to the front (back) cross-aisle or articles that are equidistant from both cross-aisles. The location of the turning points of `midpoint` can only lead to identical or longer tour lengths than obtained with `largest gap`. Nevertheless, `midpoint` also brings its advantages. While a picker can easily determine the turning points of each aisle on the go following the rules of `midpoint`, this quickly becomes difficult for `largest gap` as soon as several articles have to be collected in the same aisle.

The following actions are possible for `midpoint`:

$$E^{aisle} = \{\texttt{1pass}, \texttt{top}, \texttt{bottom}, \texttt{mgap}, \texttt{void}\} \qquad \text{and} \qquad E^{cross} = \{\texttt{00}, \texttt{11}, \texttt{02}\}.$$

Compared to the routing policy `largest gap`, the action `mgap` replaces `gap`, thus respecting the rules for the determination of the turning points of `midpoint`. Just as with `largest gap`, the actions `top` or `bottom` are used instead of `mgap` if there is only one requested article in an aisle or if the largest gap of this aisle is located at one end of this aisle.



(a) Actions within an aisle j. Note that green (blue) arcs are only valid in (non-)empty aisles. Moreover, `1pass` and `bottom` with head `END` are only valid in the right-most non-empty aisle. Between two `uu1c` states is either `mgap` or `top` or `bottom` allowed.

(b) Actions of the cross-aisles. Note that green (blue) arcs are only valid in (non-)depot cross-aisles, i.e., if $j = j_0$ ($j \neq j_0$). Dashed arcs are valid in all cross-aisles. The action from state `END` to `END` is `02` (`00`) for $j+1 \leq j_0$ ($j+1 > j_0$).

Fig. 7 Possible actions between states for the policy `midpoint`

The possible states of `midpoint` remain the same compared to `largest gap`:

$$\mathscr{S} = \{\texttt{uu1c}, \texttt{0e1c}, \texttt{000c}, \texttt{001c}, \texttt{END}\}$$

The resulting possible combinations of states and actions for `midpoint` are almost identical to the combinations of `largest gap` (compare Figs. 6 and 7). The only difference is the use of the aisle action `mgap` instead of `gap` between two states `uu1c` in Fig. 7a which differ in the definitions of the turning points.

2.6 Routing policy composite

The policy `composite` is a combination of the routing policies `traversal` and `return`. That means the picker can either traverse an aisle completely or enter the aisle and move through it until all demanded articles are collected before returning to the same end again, where the aisle was entered. These two options are possible for each aisle that needs to be visited, regardless of whether it is one of the two aisles on the edge of the tour or not.

To the best of our knowledge, there are two different definitions of `composite` in the literature. In both cases, the decision whether an aisle is visited with action `1pass`, `bottom`, or `top` is made in a greedy fashion. We first describe the two versions of the policy `composite` and then present the state space of a new version in which the distance-minimal route is generated when only allowing the actions `1pass`, `bottom`, `top`, and `void`.

Petersen (1995) introduced the routing policy `composite`. Here, the distance between the two most distant articles of two successive aisles is used as the decision criterion which action is chosen. To be more precise, in both aisles, this concerns the article that is furthest away from the picker's current position (crossing of the current aisle with back or front cross-aisle). Aisles without relevant articles are ignored. This may result in relevant successive aisles not being contiguous, but separated by irrelevant aisles. If the path resulting from applying `1pass` (`bottom` or `top`) between these two pick positions is shorter than the path resulting the other action, the former is chosen for the current aisle. If the two paths are of equal length, it is irrelevant which action is selected. According to this principle, a decision is made for each aisle on how it is passed through, starting with the aisle next to the depot. After all articles from the pick list are collected, the picker moves back to the depot through the cross-aisle where the depot is located.

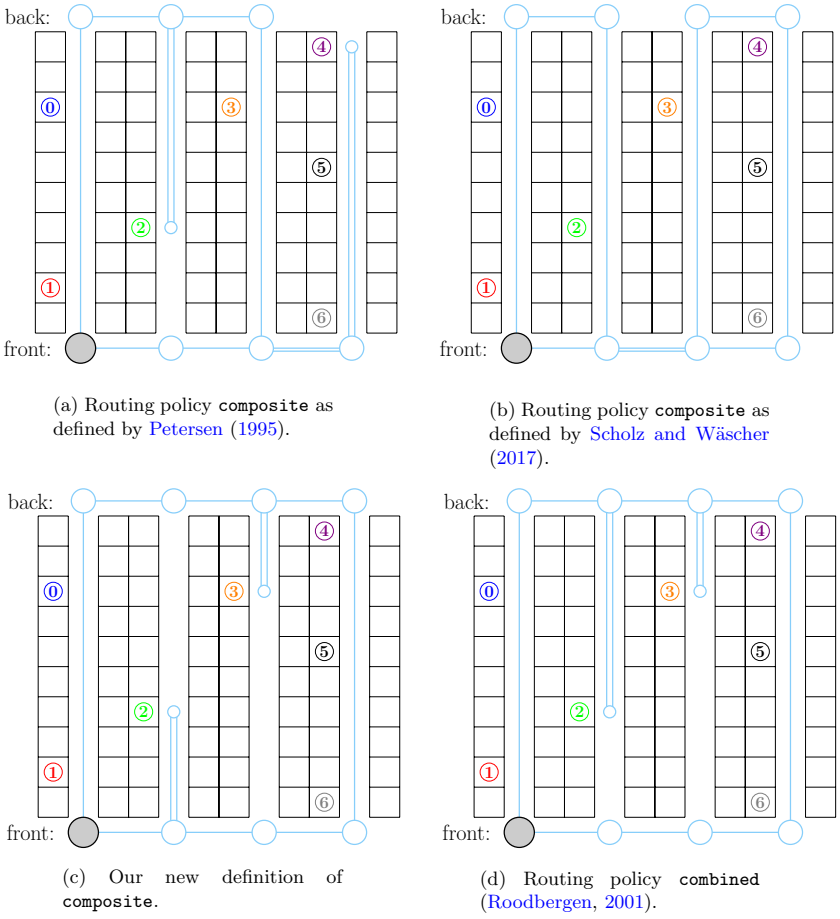
Scholz and Wäscher (2017a) define the routing policy `composite` differently. If more than half of an aisle needs to be visited to collect all demanded articles in this aisle, then the action `1pass` is chosen. However, if the picker can collect all necessary articles in the first half of the aisle, the picker follows the action `bottom` or `top` (depending on the current position) and leaves the aisle at the same end again.

It should be noted that neither of the above definitions leads to an easy-to-compute tour. In addition, the resulting tour does not follow a simple pattern that can be easily remembered. We use a third and new version of `composite` in which the actions `1pass`, `bottom`, and `top` are combined optimally. More precisely, a distance-minimal picker tour is constructed, only allowing the possible actions of these two routing policies. For this purpose, a state space is set up which contains the possible actions of `traversal` and `return`.

This procedure should not be confused with another routing policy called `combined` (Roodbergen 2001), which also combines the possible actions of `traversal` and `return` in another target-oriented manner. `combined` searches for a distance-minimal path starting at the depot and ending at the pick position of the last demanded article while collecting all other articles on the way. The way back to the depot leads through the cross-aisle of the depot. In contrast, our version of `composite` optimizes the entire picker tour and allows the collection of articles on the way back to the depot.

The state space resulting from our new definition of `composite` includes the picker tours according to the previous interpretations of `composite` as well as tours according to the `combined` policy. Thus, the corresponding picker tours dominate the others.

A visual representation of the differences in the definitions of and the routing policy `combined` can be found in Fig. 8. In each version, the same instance is used in which the distance between two neighboring pick positions as well as the distance between the first (last) pick position of an aisle and the bordering cross-aisle is one. The routing costs show clear differences for the different definitions. The picker tour has a length of 68 when constructed according to the definition of Petersen but a length of 56 results from the definition of Scholz and Wäscher. Here, the picker tour is identical to the one resulting from the routing policy `traversal`. Our new



(a) Routing policy `composite` as defined by Petersen (1995).

(b) Routing policy `composite` as defined by Scholz and Wäscher (2017).

(c) Our new definition of `composite`.

(d) Routing policy `combined` (Roodbergen, 2001).

Fig. 8 Different definitions of the routing policy `composite` in comparison with the routing policy `combined`

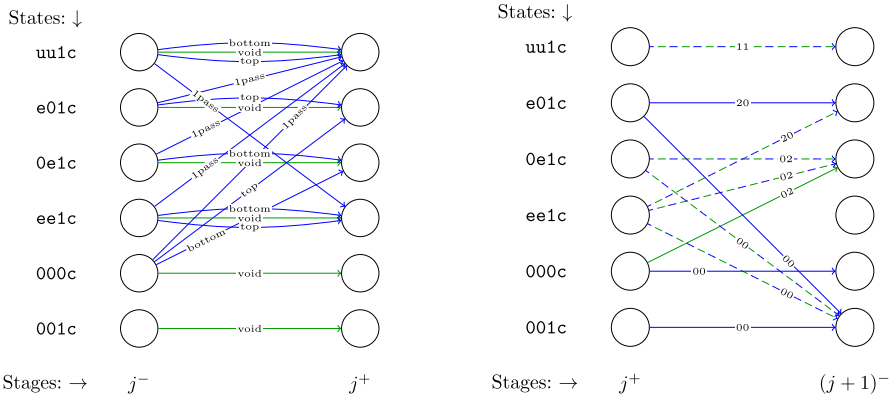
version of `composite` leads to a tour length of 48. In contrast, the picker tour resulting from `combined` has a length of 54.

Our version of `composite` has the following the sets of actions:

$$E^{aisle} = \{\texttt{1pass}, \texttt{top}, \texttt{bottom}, \texttt{void}\} \qquad \text{and} \qquad E^{cross} = \{\texttt{00}, \texttt{11}, \texttt{20}, \texttt{02}\}.$$

These two sets include all relevant actions of `traversal` and `return` for the both possible scenarios of the depot being located in the back or front cross-aisle. The possible actions of `composite` are not dependent on the position of the depot. The set of states is

$$\mathcal{S} = \{\texttt{uu1c}, \texttt{e01c}, \texttt{0e1c}, \texttt{ee1c}, \texttt{000c}, \texttt{001c}\}$$



(a) Actions within an aisle j. Note that green (blue) arcs are only valid in (non-)empty aisles.

(b) Actions of the cross-aisles. Note that green (blue) arcs are only valid in (non-)depot cross-aisles, i.e., if $j = j_0$ ($j \neq j_0$). Dashed arcs are valid in all cross-aisles.

Fig. 9 Possible actions between states for the routing policy `composite`

and contains almost all states required for the routing policy `exact`. Solely `ee2c` is missing. Furthermore, no additional state is needed for the first or last aisle of the picker tour because the rules of `composite` treat every aisle equally.

Although this routing policy already allows several different actions and needs only one state less than `exact`, the number of possible state-action combinations is substantial smaller for `composite`. Comparing Figs. 2 and 9, this difference is clearly visible.

2.7 Summary

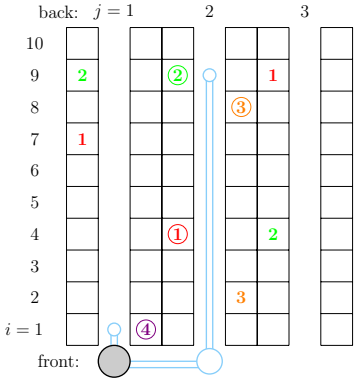
Up to now, we have established DP formulations for the SPRP restricting picker routes to the rules of the different heuristic routing policies used in practice. These DP formulations may seem dispensable, since heuristic routing policies are simple route-construction procedures. However, a recent result of Heßler and Irnich (2022b) is applicable here showing that, if properly implemented, all DPs can be constructed and solved in linear time (measured in the number of aisles of the warehouse and the number of given pick positions). Even more, the following section shows that also the more difficult variants of the problem with scattered storage can be solved with a structurally simple integer-programming formulation based on the state spaces that have been presented in this section.

3 Extensions of the state spaces and IP formulation for scattered storage

The SPRP-SS distinguishes from the basic SPRP by the additional flexibility that one can select a pick position for a requested article whenever this article is stored at two or more positions. For capturing these options, the state space must be extended (Heßler and Irnich 2024). We again present this idea as in Sect. 2 for a standard warehouse with parallel aisles and a single block layout. Compared to the SPRP, the number of stages and the states within each stage remain identical, i.e., those of the state space of Ratliff and Rosenthal. Moreover, cross-aisle actions E_j^{cross} remain identical (connecting stages j^+ and $(j+1)^-$), and also the aisle actions `1pass` and `2pass` remain unchanged. However, additional aisle actions have to be added what we explain with the help of the following example.

Example 1 An instance of the SPRP-SS with four requested articles $S = \{1, 2, 3, 4\}$ is depicted in Fig. 10a. For simplicity, we discuss the unit-demand case first, i.e., $q_s = 1$ for all $s \in S$. In aisle $j = 1$, it is possible to pick only article 4 because articles 1 and 2 are also available in other aisles. The associated new aisle action is `bottom`(1) which is the traversal from the front cross-aisle with a U-turn in cell $i = 1$. Moreover, picking only articles 1 and 4 or articles 2 and 4 is feasible leading to the additional aisle actions `bottom`(7) and `gap`$(1, 9)$ (the latter is leaving out the position $i = 7$ where article 1 is stored). These are all additional aisle actions as listed in the table in Fig. 10b. Note that any action `gap`(i, k) can be disregarded if i and k are neighboring positions with a non-maximal gap. As in the DP of Ratliff and Rosenthal, these actions are dominated by one with maximum gap.

Also in aisle $j = 2$, it is not required to collect articles 1 and 2. However, article 3 must be collected, either from position $i = 2$ or $i = 8$ making `void` and `top`(9)



(a) The warehouse and optimal picker tour.

Aisle	Type of	additional aisle actions
$j = 1$	`bottom`(i)	Cells $i \in \{1, 7\}$
	`gap`(h, i)	Cell $(h, i) = (1, 9)$
$j = 2$	`top`(i)	Cells $i \in \{4, 8\}$
	`bottom`(i)	Cells $i \in \{2, 4, 8^*\}$
	`gap`(h, i)	Cells $(h, i) \in \{(2, 8)^‡, (2, 9), (4, 9)\}$
$j = 3$	`top`(i)	Cell $i = 9$
	`bottom`(i)	Cell $i = 4$
	`void`	

(b) Additional aisle actions compared to non-scattered storage. $*$: dominated by `bottom` (4). ‡: dominated by `gap` (2 , 9).

Fig. 10 Instance of the SPRP-SS. The pick list contains four articles $S = \{1, 2, 3, 4\}$ (unit demand); pick operations are encircled

impossible. With top(4) and gap(4, 9) we can collect all three articles 1, 2, and 3. The aisle actions top(8), gap(2, 8), and gap(2, 9) can pick articles 2 and 3 but not 1. In this case, gap(2, 8) is dominated by gap(2, 9) in the sense that the latter is less costly but can collect the same articles (in the unit-demand case). Articles 1 and 3 are collected with bottom(4) and bottom(8) where the latter is dominated by the first. Lastly, with bottom(2), only article 3 can be picked.

It is possible to not enter the last aisle $j = 3$, making void a feasible option here. Moreover, additional aisle actions collecting only article 1 or article 2 are top(9) and bottom(4), respectively.

Overall, the set of aisle actions becomes aisle dependent. Hence, we denote by E_j^{aisle} the resulting set of arcs connecting stages j^- and j^+ for all $j \in J$. Several aisle actions of the same type referring to the same aisle can be considered as parallel arcs in the DP state space (in the above example, two parallel aisle actions top in aisle 2 differing in cost and articles that can be collected). Please note that, compared to the basic SPRP discussed in Sects. 2.1–2.6, the aisle actions referring to empty aisles (depicted in green in Figs. 2, 3, 4, 5, 6, 7 and 9) are now also possible for (some) non-empty aisles.

As mentioned above, the set of additional aisle actions can be reduced by dominance considerations. In the example in aisle $j = 2$, gap(2,8) is dominated by gap (2,9) because of its higher cost. This is only true in the unit-demand case because otherwise, gap(2,8) could provide more items of article 3. Likewise, bottom(8) is dominated by bottom(4) in aisle $j = 2$.

For scattered storage with general-demand, any aisle action $e \in E_j^{aisle}$ that leaves out some positions in aisle j is feasible if and only if the quantity that can be collected from this aisle together with all quantities stored in other aisles $j' \neq j$ is sufficient to fulfill the requested demand of the pick list.

In all cases (unit-demand and general-demand), we define b_{se} as the quantity of article $s \in S$ that can be collected when traversing the aisle via aisle action $e \in E_j^{aisle}$. Cross-aisle actions have zero supply $b_{se} = 0$ for all $e \in E_j^{cross}$ and $j \in J$. For a given article $s \in S$, we denote the set of edges with a non-negative quantity b_{se} by E_s.

Example 2 (continued from Example 1) For the SPRP-SS instance depicted in Fig. 10, we now assume that each position stores three units of each articles of the respective indicated type (the general-demand case). Then, overall, nine, nine, six, and three articles 1, 2, 3, and 4 are stored in the warehouse, respectively. Any demand $q = (q_1, q_2, q_3, q_4) \leq (9, 9, 6, 3)$ can be retrieved from a picker tour. The picker tour depicted in Fig. 10a can collect (3, 3, 6, 3) units. This picker tour uses in aisle $j = 1$ the aisle action bottom(1) with positive supply value $b_{se} = 3$ for $s = 4$ and $e = \texttt{bottom}(1) \in E_1^{aisle}$. In aisles $j = 2$, the aisle action bottom(9) has positive supply values $b_{1e} = 3$, $b_{2e} = 3$, and $b_{3e} = 6$ for $e = \texttt{bottom}(9) \in E_2^{aisle}$. In aisle $j = 3$, the aisle action void has no positive supply values.

We can now formalize the DP-based formulation for the SPRP-SS. Recall that we have stages $1^-, 1^+, 2^-, 2^+, \ldots, m^-, m^+, (m+1)^-$. Let V denote the set of

all states of all stages, i.e., the vertices of state space. In particular, there are two distinct states, the initial state $o = \texttt{000c}$ at stage 1^- and the final state is d at stage $(m+1)^-$. The latter state is either $d = \texttt{001c}$ or $d = \texttt{END}$ depending on the routing policy, see Sect. 2. Moreover, let E denote the (disjoint) union of all aisle and cross-aisle actions, i.e.,

$$E = \bigcup_{j=1}^{m} \left(E_j^{aisle} \cup E_j^{cross} \right).$$

For each $e \in E$, let c_e be the length of the corresponding part of a picker tour described by e. The discrete linear formulation uses binary variables x_e for all $e \in E$ to indicate whether the optimal picker tour contains the respective part of a walk described by e. Heßler and Irnich (2024) propose the following model:

$$\min \sum_{e \in E} c_e x_e \tag{1a}$$

$$\text{subject to} \quad \sum_{e \in \delta^+(\sigma)} x_e - \sum_{e \in \delta^-(\sigma)} x_e = \begin{cases} +1, & \text{if } \sigma = o \\ -1, & \text{if } \sigma = d \\ 0, & \text{otherwise} \end{cases} \qquad \forall \sigma \in V \tag{1b}$$

$$\sum_{e \in E_s} b_{se} x_e \geq q_s \quad \forall s \in S \tag{1c}$$

$$x_e \in \{0, 1\} \quad \forall e \in E \tag{1d}$$

The objective (1a) is the minimization of the length of the resulting picker tour. The flow-conservation constraints (1b) ensure that the selected aisle and cross-aisle actions together describe a o-d-path in the extended state space (V, E). For any state $\sigma \in V$, $\delta^+(\sigma)$ and $\delta^-(\sigma)$ denote the set of arcs leaving and entering state σ, respectively. Demand fulfillment, i.e., that the requested number q_s of articles $s \in S$ can be collected with the constructed tour, is guaranteed by (1c). The domain of the flow variables x_e is given by (1d). Note that (1a), (1b), and (1d) is the standard model of an origin–destination shortest-path problem.

Example 3 (continued from Example 2) We pursue the example depicted in Fig. 10 and this time use the different routing policies presented in Sect. 2. Here, we have the seven stages 1^-, 1^+, 2^-, 2^+, 3^-, 3^+, and 4^-.

For the routing policy `exact`, the initial state is $o = \texttt{000c}$ at stage 1^-, and the final state is $d = \texttt{001c}$ at stage 4^-. The optimal solution depicted in Fig. 10a can be described by the corresponding sequence

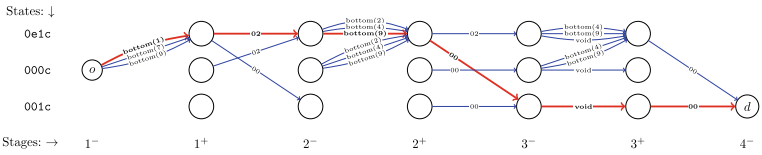


Fig. 11 State space for the SPRP-SS instance depicted in Fig. 10a and routing policy `return`. The shortest *o*-*d*-path corresponding to the optimal picker tour is marked in red/thick. Note that only paths collecting all articles $S = \{1, 2, 3, 4\}$ are feasible

$$(\texttt{000c}, \texttt{0e1c}, \texttt{0e1c}, \texttt{0e1c}, \texttt{001c}, \texttt{001c}, \texttt{001c})$$

of states. The optimal control is $(\texttt{bottom}(1), \texttt{02}, \texttt{bottom}(9), \texttt{00}, \texttt{void}, \texttt{00})$.

For the routing policy `return`, the exact same control $(\texttt{bottom}(1), \texttt{02}, \texttt{bottom}(9), \texttt{00}, \texttt{void}, \texttt{00})$ is also optimal (feasibility can be tested with the help of Fig. 11), leading to the same sequence of states that the *o*-*d*-path travels through and the same final picker tour.

For the routing policies `traversal`, `largest gap`, and `midpoint`, the picker tour depicted in Fig. 10a is not feasible. Applying the same control $(\texttt{bottom}(1), \texttt{02}, \texttt{bottom}(9), \texttt{00}, \texttt{void}, \texttt{00})$ is infeasible, since the action $\texttt{bottom}(1)$ would let the path transit through the state `END` where it remains with action `02`. From state `END`, the action $\texttt{bottom}(9)$ is not allowed, see Figs. 4a, 6a, and 7a.

Finally, we consider the routing policy `composite` with the three different underlying definitions explained in Sect. 2.6. All three definitions lead to the picker tour depicted in Fig. 10a. The control $(\texttt{bottom}(1), \texttt{02}, \texttt{bottom}(9), \texttt{00}, \texttt{void}, \texttt{00})$ consists of two `bottom` actions and one `void` action within the aisles. In general, these actions are possible when applying `composite`. Regarding our new definition of `composite`, the state space constructed from the actions of Fig. 9 allows the solution shown in Fig. 10a. Following the definitions of `composite` by Petersen (1995) and Scholz and Wäscher (2017a), this picker tour is also feasible.

Algorithm 1 summarizes the solution approach described so far. The construction of the state space of the SPRP in Step 1 depends on the warehouse layout, the routing policy `plc` as described in Sects. 2.1–2.6, and the given positions of the SKUs. Note that the destination vertex d depends on the routing policy `plc` (Step 2, state `001c` or `END`). The computation of additional aisle actions in Step 3 has been explained in Sect. 3. In Steps 5 and 6, the costs c_e and the supply values b_{se} depend on the given positions of the SKUs and the warehouse layout. The final picker tour is a closed walk w over all parts that result from the chosen aisle actions, i.e., variables x_e with value one of the IP-formulation (1) solved in Step 7. The efficient computation of the walk w constructed in Step 8 is explained in (Ratliff and Rosenthal 1983, Sect. 4).

Algorithm 1 Exact Algorithm for SPRP-SS

Input : SPRP-SS instance, i.e., layout $(m, C,$ distances$)$, routing policy `plc`, demands q_s, and positions of SKUs with supply quantities
1 Construct DP state space (V, E^{SPRP}) for SPRP
2 Set origin o and destination d
3 Construct additional aisle actions E^{SS} for SPRP-SS
4 Set $E = E^{SPRP} \cup E^{SS}$
5 Compute c_e for all $e \in E$
6 Compute b_{se} for all $s \in S$ and $e \in E$
7 Solve formulation (1) over (V, E, o, d) with a MIP solver
8 Construct picker tour w from MIP solution
Output: Picker tour w

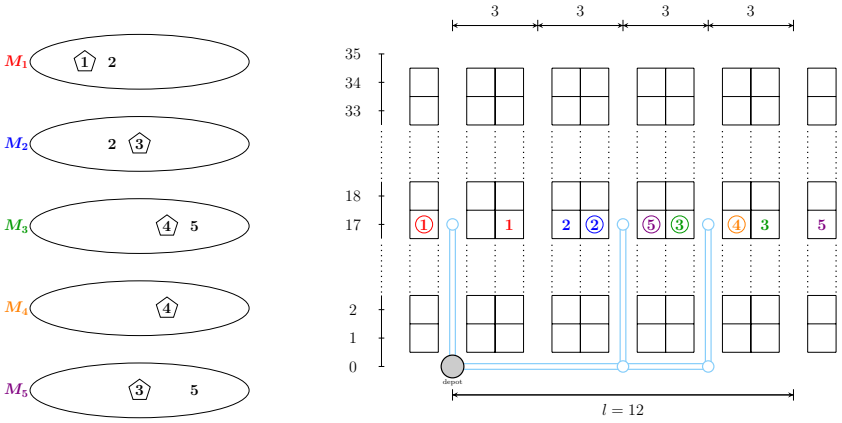
4 NP-hardness of the SPRP-SS

As mentioned above, Weidinger (2018, Theorem 1) has shown that computing a distance-minimal picker tour (policy `exact`) in a one-block parallel-aisle warehouse for the SPRP-SS is NP-hard. We now show an even stronger result, i.e., the problem remains NP-hard even if a shortest picker tour for one of the commonly used heuristic routing policies has to be determined. The following proposition summarizes our findings about the computational complexity of the SPRP-SS.

Proposition 1 *The SPRP-SS is strongly NP-hard for the routing policies* `traversal`, `return`, `midpoint`, `largest gap`, *and* `composite` *considering a one-block parallel-aisle warehouse.*

Proof The proof is analog to (Weidinger 2018, Theorem 1) through a reduction from the hitting set problem that is strongly NP-complete. We show that the existence of a hitting set is equivalent to the existence of a picker tour of a certain maximal length.

The hitting set problem is defined as follows (Garey and Johnson 1979): Given a finite set T, a positive integer $k \leq |T|$, and a set of subsets $\{M_1, \ldots, M_n\}$ with $M_i \subseteq T$



(a) Hitting set problem with $T = \{1, 2, 3, 4, 5\}$, $k = 3$, and sets $M_1, \ldots, M_5$ as shown. The solution is the hitting set $H = \{1, 3, 4\}$.

(b) The corresponding picker routing problem with $m = 5$ aisles and $C = 34$ cells per aisle. Article i is stored in aisle j if and only if $j \in M_i$. The length of the plotted optimal picker route for the policies `return`, `composite`, and `midpoint` is $120 \leq kC + 2l = 126$. Aisles 1, 3, and 4 are entered.

Fig. 12 Example for the transformation between the hitting set problem and the picker routing problem. The hitting set corresponds to the entered aisles

for all $i \in \{1, \ldots, n\}$, find a set $H \subseteq T$ such that $|H| \leq k$ and $H \cap M_i \neq \emptyset$ for all $i \in \{1, \ldots, n\}$.

Given an instance $\langle T, k, \{M_1, \ldots, M_n\}\rangle$ of the hitting set problem, a corresponding instance of the SPRP-SS can be constructed as follows. We consider a single-block warehouse with $m = |T|$ aisles and $C = 2l + 2|T|$ cells per aisle. The depot is located at the front-end of the leftmost aisle. Let the (vertical) distance between two neighboring cells be 1 and the (horizontal) distance between two neighboring aisles be 3. Then, $l = 3(|T| - 1)$ is the distance between the first and last aisle (see Fig. 12). All n articles that need to be collected are stored in the lower middle cell $c = l + |T| + 1$. Article i is stored in aisle j if and only if $j \in M_i$. The figure illustrates how the SPRP-SS instance is constructed. We now prove that the existence of a hitting set of size $|H| \leq k$ is equivalent to the existence of a picker tour of maximal length $k(C + 1) + 2l$.

First, we consider the routing policies `return`, `composite`, and `midpoint`.

'⇒': Given a hitting set of size $|H| \leq k$. According to the routing policy, we construct the corresponding picker tour that enters only the aisles $i \in H$, i.e., not more than k aisles. For the policies `return`, `composite`, `midpoint`, and the constructed instance, this picker tour consist of `bottom` actions only for collecting articles. The traveled distance is upper bounded by

$$\underbrace{|H|C}_{\texttt{bottom actions}} + \underbrace{2l}_{\text{cross-aisle actions}} \overset{|H|\leq k}{\leq} k(C+1)+2l, \tag{2}$$

where the first summand is the distance of $|H|$ `bottom` traversals and the second summand is an upper bound on the horizontal cross-aisle traversal distance. This shows the first direction.

'⇐': Given a picker tour of maximal length $k(C+1)+2l$ collecting all articles. It must be shown that the picker tour enters not more than k different aisles. We prove this direction by contradiction: If $k+1$ or more aisles were entered, at least one article was collected in each aisle. This results in a picker tour of length not shorter than

$$\underbrace{(k+1)C}_{\texttt{bottom actions}} + \underbrace{2\cdot 3\cdot k}_{\text{cross-aisle actions}} > kC + C = kC + 2l + 2|T| \overset{k\leq|T|}{\geq} kC + 2l + k = k(C+1)+2l \tag{3}$$

contradicting with the maximal tour length of $k(C+1)+2l$. Consequently, the picker collects the articles from not more than k different aisles. Defining the hitting set H as the set of all entered aisles, we have proven this direction, too.

Second, we consider the routing policy `traversal`.

'⇒': The shape of the `traversal` picker tour that enters only the aisles $i \in H$ depends on whether $|H|$ is even or odd. If $|H|$ is even, the length is at most $|H|(C+1)+2\,l \leq k(C+1)+2\,l$, where the first summand is the distance of $|H| \leq k$ `1pass` traversals and the second summand is an upper bound on the horizontal cross-aisle traversal distance. If $|H|$ is odd, the picker tour consists of $k-1$ `1pass` traversals and one `bottom` traversal, which amounts a shorter tour length of $(|H|-1)(C+1)+C+2l \leq k(C+1)+2l$.

'⇐': As in the first case, if $k+1$ or more aisles were entered, the contradiction results from replacing (3) by

$$\underbrace{k(C+1)}_{\texttt{1pass actions}} + \underbrace{C}_{\texttt{bottom actions}} + \underbrace{2\cdot 3\cdot k}_{\text{cross-aisle actions}} > k(C+1)+C \overset{C=2l+2|T|}{>} k(C+1)+2l. \tag{4}$$

Third, we consider the routing policy `largest gap`.

'⇒': The `largest gap` picker tour that enters only the aisles $i \in H$ has the following shape: two `1pass` traversals in the first and last non-empty aisle and $|H|-2$ `largest gap` traversals that are actually `bottom` traversals because only one position (at x) stores articles in a non-empty aisle. The distance of this optimal picker tour is at most

$$2(C+1)+(|H|-2)C+2l \overset{|H|\leq k}{\leq} 2(C+1)+(k-2)C+2l \leq k(C+1)+2l, \tag{5}$$

where the first summand is the two traversals in the first and last non-empty aisle, the second summand is the distance of $|H|-2$ `bottom` traversals, and the third

summand is an upper bound on the horizontal cross-aisle traversal distance. This proves the first direction.

'⇐': As in the first case, if $k+1$ or more aisles were entered, the contradiction results from replacing (3) by

$$\underbrace{2(C+1)}_{\texttt{1pass actions}} + \underbrace{(k-1)C}_{\texttt{gap actions}} + \underbrace{2 \cdot 3 \cdot k}_{\text{cross-aisle actions}} > k(C+1) + C \overset{C=2l+2|T|}{>} k(C+1) + 2l.$$

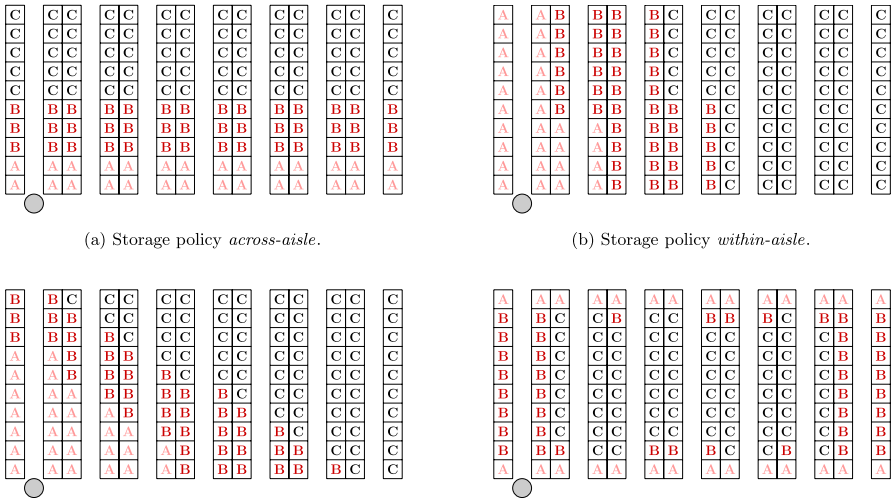
□

5 Class-based storage policies

de Koster et al. (2007) categorize several methods for the assignment of articles to pick positions. One type of these methods are the *class-based storage policies*, where each article is assigned to a pick position according to its value, e.g., the turnover rate, see Sect. 1.2. For this purpose, articles are classified into one of the classes A, B, or C based on their value. As commonly assumed, 20% of all articles with the highest value are assigned to class A, the next 30% are assigned to class B, and the remaining 50% of the articles belong to class C. We often observe then that classes A, B, and C are composed of approximately 80%, 15%, and 5% of the articles, respectively.

The pick positions of the warehouse are grouped into three zones (one zone for each class), and the different class-based storage policies vary in how the zones are defined. As in (Petersen and Schmenner 1999), we consider the typical class-based policies *across-aisle*, *within-aisle*, *perimeter*, and *diagonal*, for which the corresponding distribution schemes are illustrated in Fig. 13. As a counterpart to the class-based policies, we include the storage policy *uniformly distributed* in the evaluation, which has already evolved into the most common storage policy in the last millennium (Petersen and Schmenner 1999, p. 483). For the latter policy *uniformly distributed*, the articles of all three classes are distributed randomly in the warehouse regardless of the classification.

The policies *across-aisle*, *within-aisle*, and *diagonal* arrange the zones in the warehouse such that zone A is positioned closest to the depot. Zone B follows after zone A, and zone C generally comprises pick positions that are farthest from the depot. The specific assignment of the pick positions to the zones depends on the distance measure used. In policy *across-aisle*, zone A includes the pick positions of each aisle that are closest to the cross-aisle of the depot. The remaining pick positions are allocated to zones B and C according to the same principle. In policy *within-aisle*, pick positions are assigned to zones aisle by aisle and pick position by pick position (as a secondary criterion): Starting with the depot aisle and the position closest to the depot cross-aisle, the remaining pick positions are assigned to the zones. When zone A (B) comprises sufficiently many pick positions, the next pick position is assigned to zone B (C). In policy *diagonal*, all pick positions are sorted by ascending distance to the depot in a first step. Afterwards, the first 20% of the



(a) Storage policy *across-aisle*.

(b) Storage policy *within-aisle*.

(c) Storage policy *diagonal*.

(d) Storage policy *perimeter*.

Fig. 13 Class-based storage policies

pick positions are allocated to zone A, and the remaining 30% and 50% of the pick positions are added to zone B and C, respectively.

In policy *perimeter*, the warehouse is split up into zones from the outside (zone A) to the inside (zones B and C): Zone A comprises the pick positions around the perimeter of the warehouse, while zone C stores articles in the center of the warehouse. Zone B contains the remaining pick positions in between, arranged in a ring-shaped fashion. Overall, pick positions are assigned to zones A to C by moving from the outside to the inside of the warehouse. To exactly respect the predefined proportions of the zones (20%, 30%, and 50%), we used the following procedure for generating the three zones A, B, and C:

1. *Select alternately the outer horizontal and vertical layer that are not yet assigned:*
   The horizontal layer k includes the kth and the kth-last pick position of each aisle. In contrast, the vertical layer k includes all pick positions in the kth and the kth-last aisle. The selection of layers is then: horizontal layer 1, vertical layer 1, horizontal layer 2, vertical layer 2 etc.
2. *Assign zones to the (unassigned) pick positions of the chosen layer in the order A, B, C:*
   If all unassigned pick positions of the layer can be added to the considered zone without exceeding the given size, they are assigned to the zone. Otherwise, if only a portion of the pick positions of the layer can be assigned to the zone, randomly select pick positions from the layer and assign them to the considered zone until the given size is reached. In the latter case, repeat the procedure with the remaining unassigned pick positions of the layer and the next zone.

6 Computational study

In this section, we present the results of our extensive computational study and analyze which combinations of routing and storage policies are beneficial for a given warehouse layout and length of order list. The state space and the resulting IP formulation are implemented in C++ and compiled in release mode into a 64-bit single-thread code under MS Visual Studio 2017. We use the callable library `CPLEX` 20.1 for optimizing the IP formulation (1a)–(1d). The default values are kept for all parameters except from setting the number of threads to one. The computational study was executed on a 64-bit Microsoft Windows 10 computer with an Intel® Core™ i7-5930k CPU clocked at 3.5 GHz and 64 GB of RAM.

6.1 Instances

To the best of our knowledge, there are no benchmark instances combining scattered storage with class-based storage policies publicly available. Therefore, we generated an extensive set of instances for our computational study. We consider a single-block warehouse with parallel aisles and two bordering cross-aisles, see Figs. 10a, 12b and 13. The depot is located in the front cross-aisle at the level of the first aisle from the left, and the distance between the depot and the bottom cell in the first aisle is 1 unit. Likewise, the distance between two neighboring pick positions of an aisle is always 1 unit. The distance between two neighboring aisles is 3 units. Finally, switching between consecutive aisles amounts to 5 units (first cell to first cell or last cell to last cell).

For the entire computational study, we fixed the warehouse layout to $m = 10$ aisles with $C = 40$ pick positions per aisle. At each pick position, there are two racks from which stored articles can be collected (left- and right-hand side are irrelevant for distance calculations). In total, there are 800 pick positions in the warehouse. We assume the articles to be rather small so that up to ten different articles can be stored at one pick position at the same time. Furthermore, we presume that a sufficiently large quantity of each article is available at a pick position so that each line of the pick list can be collected from one pick position (the unit-demand case, see Sect. 1.2).

The number of different articles (SKUs) stored in the warehouse is fixed to $n = 1000$ for all instances to ensure comparability. Hence, there are $n_A = 200$ class-A, $n_B = 300$ class-B, and $n_C = 500$ class-C articles.

For the storage policy *uniformly distributed*, scattering is performed as follows. If the global scatter factor is α there must be $n\alpha$ different pairs (a, p) of article a and pick position p. More precisely, for a given scatter factor setting $(\alpha_A/\alpha_B/\alpha_C)$, we generate $n_A\alpha_A + n_B\alpha_B + n_C\alpha_C = n\alpha$ pairs in the following way: First, each article a is assigned to a randomly chosen pick position p (while making sure that no position contains more than 10 different articles). This guarantees that every article is available and stocked at least once.

Table 2 Parameters used for the generation of SPRP-SS instances

Parameter	Value(s)
Number of aisles m	10
Number of pick positions C (per aisle)	40
Max. storage capacity per pick position	10 articles
Number of different articles n	1000
Relative sizes of zones A, B, and C	20%, 30%, and 50%
Overall scatter factor α	2
Number of different pairs of article and pick position αn	2000
Scatter factor settings ($\alpha_A/\alpha_B/\alpha_C$)	(6/1/1), (1/1/3), (2/2/2)
Number of articles to pick = length of pick list	5, 15, 50
Distribution of A-, B-, and C-articles in pick list	80%, 15%, and 5%

Second, for each class $X \in \{A, B, C\}$, we generate $n_X(\alpha_X - 1)$ additional pairs (a, p) (excluding duplicates and again making sure that no position contains more than 10 different articles).

Third, for class-based storage policies (*across-aisle*, *within-aisle*, *perimeter*, and *diagonal*), we use the same instance generation mechanism but extend the procedure by the following additional step: The warehouse is divided into three zones, one zone for each class A, B, and C depending on the storage policy and according to the sizes of the zones, see Sect. 5. Hence, zone A comprises 20%, zone B 30%, and zone C 50% of all pick positions, respectively. Next, the pairs (a, p) generated in the second step are randomly relocated to feasible positions p' where the position p' must belong to the class of article a. As a result, the new set of pairs (a, p') consistently stores all articles according to the given class-based storage policy.

Fourth and finally, the pick list must be created. The length of the pick list is fixed to 5, 15, or 50. Given the length, we first assign each order line with a probability of 80%, 15%, and 5% to class A, B, or C, respectively. Next, we randomly choose an article from the class (rejecting duplicate articles in the pick list).

Table 2 summarizes the parameters used for generating instances for the computational experiments. This includes the different scatter factor settings and lengths of pick lists. For each parameter combination, we generated 50 instances. In total, this gives $5 \cdot 3 \cdot 3 \cdot 50 = 2250$ instances (five storage policies, three scatter factor settings, and three lengths of pick lists) to be analyzed with six routing policies (see Sect. 2). The instances are online available at https://logistik.bwl.uni-mainz.de/research/benchmarks/.

6.2 Computational performance

As noted in Sect. 4, solving the SPRP-SS is NP-hard, even when heuristic routing policies are used. However, all instances of the computational study can be solved within milliseconds. Table 3 shows the average and maximum computation times (in milliseconds) for the different routing policies and pick list lengths. Times include the construction of the IP model (which is negligible) and the actual solution

Table 3 Average and maximum computation times in milliseconds for different routing policies and order sizes

		Average time			Max. time		
Order size		5	15	50	5	15	50
Routing policy	exact	6.60	15.57	72.78	48.64	227.38	634.91
	composite	3.27	5.47	11.38	26.53	112.33	160.16
	largest gap	1.54	2.20	7.93	14.43	20.57	141.76
	midpoint	1.40	1.78	4.03	5.86	16.97	75.66
	return	1.42	1.59	1.75	15.21	10.44	16.77
	traversal	2.01	1.83	1.29	16.01	40.92	7.87

time consumed by the MIP solver (CPLEX is restricted to use a single CPU thread only). One can see that more complex routing policies like exact or composite lead to longer computation times than fairly simple routing policies like return or traversal. The reason for this are the differently sized state spaces, see Sect. 2. While complex routing policies allow (almost) all states and many different actions within an aisle for the different states, the state space of a simple routing policy is rather small and sparse. The order size and the length of the pick list also influence the computation time (note that in the unit-demand case, order size equals pick list length; we will use the shorter term 'order size' in the following). In particular, if the considered routing policy allows many different actions within an aisle for many different states, a larger order size can result in a tenfold increase in runtime. In contrast, for small and sparse state spaces, this effect is very small or disappears completely. Despite a maximum runtime of 634.91 milliseconds for an instance with a pick list length of 50 and routing policy exact, the computation times are very short overall. Therefore, we do not go into further algorithmic details and neglect a more in-depth analysis of the runtimes. Instead, we will focus on evaluating different routing and storage policy settings in the following.

6.3 Evaluation of combinations of routing and storage policies

Next, we evaluate combinations of routing policies and storage policies for the SPRP-SS. To this end, each of the earlier introduced routing policies is combined with every storage policy. In addition, both the scatter factor settings ($\alpha_A/\alpha_B/\alpha_C$) and the order sizes are varied.

Table 4 provides aggregated results for each combination. (For the sake of readability, some routing and storage policies are abbreviated in the tables: *uniformly distributed* is denoted as *unif. distr.*, largest gap as gap, *across-aisle* as *across*, and *within-aisle* as *within*.) The upper part of the table shows the average tour lengths. A striking observation are the large differences in the tour lengths. While the combination of return and *uniformly distributed* leads to an average tour length of 351.31, the policies exact and *within-aisle* induce tours that are nearly 60% shorter with a mean length of 149.04. Even if the optimal routing

Table 4 Average picker tour length for all combinations of routing policies and storage policies

		Storage policy				
		unif. distr.	*within*	*across*	*perimeter*	*diagonal*
Routing policy	`exact`	233.27	149.04	163.31	165.68	162.20
	`composite`	238.82	151.61	164.30	185.46	163.99
	`largest gap`	256.91	165.93	197.24	174.61	186.83
	`midpoint`	268.50	169.86	198.80	176.36	189.54
	`return`	351.31	204.38	197.45	287.40	204.45
	`traversal`	290.00	181.35	284.22	249.73	227.52
Deviation	`composite/exact`	2.38%	1.72%	0.61%	11.93%	1.10%
	`midpoint/largest gap`	4.51%	2.37%	0.79%	1.00%	1.45%

policy `exact` is replaced by `composite`, the tour hardly gets any longer on average. On the contrary, for a given storage policy, the average tour lengths differ considerably for all heuristic routing policies. Similar observations can be made when the routing policy is fixed and the storage policy is varied. Thus, we conclude that both the choice of the routing policy and the storage policy have a major impact on the tour length.

When comparing the routing policies in Table 4, it is noticeable that there are two policies each that behave very similarly. These are `composite` and `exact` as well as `midpoint` and `largest gap`. The lower part of the table shows that the deviation of `composite` (`midpoint`) from `exact` (`largest gap`) is less than 5% for every storage policy, except for the single combination with *perimeter* (with a deviation of nearly 12%). As the tour lengths hardly differ for these routing policies, we exclude `composite` and `midpoint` from further analyses. Routing policy `exact` is preferred over `composite` to be able to compare optimal rule-based and minimum-length tours. Besides, as already discussed in Sect. 2.6, the heuristic routing policy `composite` has rather difficult rules. Thus, the other routing policies are more attractive, because they are easier to remember and are expected to lead to fewer human errors. The rules for `largest gap` and `midpoint` only differ in the turning point within an aisle. Tours resulting from `largest gap` are always shorter or as long as the tour resulting from `midpoint` (this known result (Hall 1993) for the non-scattered case directly transfers to the case of scattering articles). Therefore, we limit the further evaluation to `largest gap` and disregard `midpoint` in the following.

Table 5 shows detailed low-level results. For each combination of a routing and storage policy, average tour lengths are listed per scatter factor setting and order size. In addition, for each combination of routing policy, scatter factor setting and order size, the storage policies leading to the shortest average tour length (min) are given. Vice versa, for each storage policy, the best routing policy is given per setting (in this case, we exclude the routing policy `exact` from the min-grading as the optimal routing always generates the shortest average tour lengths; policy `exact` is shaded in italic in Table 5). In this spirit, the routing policy `exact`

Table 5 Break down of average tour lengths on the level of scatter factor settings and order sizes

			unif. distr.	*perim-eter*	*across*	*within*	*diagonal*	*min*
	(a) Order size 5							
Scatter factor setting	(1/1/3)	exact	*155.20*	*139.08*	*70.60*	*89.12*	*72.52*	*across*
		gap	167.36	141.08	139.44	90.24	114.92	*within*
		return	220.80	186.92	**70.60**	110.36	72.88	*across*
		traversal	214.56	158.76	200.88	90.40	127.16	*within*
		min	gap	gap	return	gap	return	
	(2/2/2)	exact	*130.56*	*127.84*	*81.52*	*87.64*	*95.48*	*across*
		gap	141.16	138.84	130.76	104.76	126.44	*within*
		return	166.36	169.12	**83.04**	95.08	97.12	*across*
		traversal	171.24	166.08	144.20	110.68	141.44	*within*
		min	gap	gap	return	return	return	
	(6/1/1)	exact	*98.12*	*93.48*	*98.04*	*92.68*	*94.76*	*within*
		gap	112.28	110.76	109.84	109.60	114.76	*within*
		return	106.16	100.60	108.72	**100.08**	100.64	*within*
		traversal	115.60	114.52	114.80	115.08	117.84	*perimeter*
		min	return	return	return	return	return	
	(b) Order size 15							
Scatter factor setting	(1/1/3)	exact	*278.88*	*152.40*	*137.00*	*114.76*	*130.12*	*within*
		gap	301.16	152.44	189.72	**122.76**	160.80	*within*
		return	447.40	266.56	137.80	156.92	134.56	*diagonal*
		traversal	373.24	241.12	385.12	131.24	212.76	*within*
		min	gap	gap	return	gap	return	
	(2/2/2)	exact	*214.08*	*164.32*	*159.12*	*152.60*	*153.68*	*within*
		gap	230.76	170.96	192.68	**162.72**	178.60	*within*
		return	328.08	254.72	170.00	211.92	168.64	*diagonal*
		traversal	291.64	242.00	297.84	189.60	238.40	*within*
		min	gap	gap	return	gap	return	
	(6/1/1)	exact	*160.64*	*154.08*	*166.96*	*151.44*	*167.08*	*within*
		gap	172.32	169.20	178.64	**165.00**	175.72	*within*
		return	215.08	214.64	221.24	198.04	228.68	*within*
		traversal	206.84	205.04	208.04	199.80	208.08	*within*
		min	gap	gap	gap	gap	gap	

Table 5 (continued)

			unif. distr.	*perim-eter*	*across*	*within*	*diagonal*	*min*
	(c) Order size 50							
Scatter factor setting	(1/1/3)	`exact`	*434.24*	*181.80*	*210.68*	*162.88*	*201.00*	*within*
		`gap`	502.32	**181.80**	248.32	187.12	218.72	*perimeter*
		`return`	706.40	555.04	220.56	241.64	231.24	*across*
		`traversal`	462.88	440.08	464.00	197.16	292.24	*within*
		min	`traversal`	`gap`	`return`	`gap`	`gap`	
	(2/2/2)	`exact`	*359.48*	*232.68*	*257.28*	*240.60*	*263.80*	*perimeter*
		`gap`	396.68	**233.52**	280.60	270.76	290.20	*perimeter*
		`return`	562.68	452.56	298.72	354.12	315.60	*across*
		`traversal`	427.24	369.04	397.80	283.24	353.84	*within*
		min	`gap`	`gap`	`gap`	`gap`	`gap`	
	(6/1/1)	`exact`	*268.20*	*245.48*	*288.56*	*249.64*	*281.32*	*perimeter*
		`gap`	288.12	**272.92**	305.16	280.40	301.28	*perimeter*
		`return`	408.80	386.44	466.36	371.28	490.72	*within*
		`traversal`	346.72	310.92	345.32	314.92	355.96	*perimeter*
		min	`gap`	`gap`	`gap`	`gap`	`gap`	

is rather used as an optimal reference solution, not as one of the compared routing policies for the subsequent evaluations.

We first concentrate on the global findings, before we analyze each class-based storage policy separately concerning the different routing policies and the variation of the scatter factor setting and the order size. In both cases, Table 5 is the basis for the evaluation, and Table 6 provides complementing data summarized across all order sizes.

General observations can be summarized as follows:

- Storage policy *uniformly distributed* performs worse compared to the class-based storage policies: It generates the longest picker tours for nearly all settings with scatter factor setting (1/1/3) and (2/2/2), see Table 5. Only for (6/1/1), *uniformly distributed* can keep up and is in the midfield compared to the other storage policies. This can also be seen in Table 6, which summarizes the average tour lengths across all order sizes. These observations can be explained by the fact that an order consists of 80% A-articles, and as *uniformly distributed* does not take advantage of the classification of the articles, it is most advantageous to scatter only the articles that appear most frequently in

Table 6 Average picker tour lengths summarized across all order sizes

			unif. distr.	*perimeter*	*across*	*within*	*diagonal*	*subtotal*
Scatter factor setting	(1/1/3)	`exact`	*289.44*	*157.76*	*139.43*	*122.25*	*134.55*	*168.69*
		`gap`	323.61	158.44	192.49	**133.37**	164.81	**194.55**
		`return`	458.20	336.17	142.99	169.64	146.23	250.65
		`traversal`	350.23	279.99	350.00	139.60	210.72	266.11
		Subtotal	355.37	233.09	206.23	141.22	164.08	
	(2/2/2)	`exact`	*234.71*	*174.95*	*165.97*	*160.28*	*170.99*	*181.38*
		`gap`	256.20	181.11	201.35	**179.41**	198.41	**203.30**
		`return`	352.37	292.13	183.92	220.37	193.79	248.52
		`traversal`	296.71	259.04	279.95	194.51	244.56	254.95
		Subtotal	285.00	226.81	207.80	188.64	201.94	
	(6/1/1)	`exact`	*175.65*	*164.35*	*184.52*	*164.59*	*181.05*	*174.03*
		`gap`	190.91	**184.29**	197.88	185.00	197.25	**191.07**
		`return`	243.35	233.89	265.44	223.13	273.35	247.83
		`traversal`	223.05	210.16	222.72	209.93	227.29	218.63
		Subtotal	208.24	198.17	217.64	195.66	219.74	

an order. We disregard the storage policy *uniformly distributed* in the following analysis and concentrate on the class-based storage policies.

- Looking at the performance of the different scatter factor settings, it can be stated that, for every order size, the scattering of C-articles leads to the shortest picker tours. We indicate the best result per order size and scatter factor setting by using bold font in Table 5. Furthermore, each routing policy also generates the shortest picker tours for each order size, if scatter factor setting (1/1/3) is applied. Comparing the tours generated by the different routing policies per scatter factor setting and order size, it is noticeable that the scatter factor settings behave differently. For scatter factor setting (1/1/3), the tour lengths resulting from the different routing policies vary considerably. In contrast, if only the A-articles are scattered, the choice of the routing policy only has a minor influence on the tour length. While A-articles are stored at beneficial pick positions close to the depot, C-articles are located far from the depot. Thus, an A-article in the order list normally implies a shorter supplementary path segment in the picker tour than a C-article. And similarly, scattering A-articles usually saves less distance than scattering C-articles, since the choice of one or several alternative pick positions within zone A potentially leads to smaller savings than in zone C.
- Comparing the different order sizes, there are particularly poor and good routing policies. With an order size of 5, in eleven of 15 combinations, the routing policy `traversal` leads to the longest tours on average. However, for order size of 50, the situation changes and `return` generates the longest picker tours in most of the combinations. The reason is that `return` is more suitable for smaller order sizes, while `traversal` is advantageous for larger order sizes. If only a few pick positions need to be visited, it is shorter to enter the aisles only from the front cross-aisle. If many positions that are distributed across many aisles need to be visited, traversing each aisle leads to shorter picker tours than entering and leaving each aisle from the same end. Looking at best-performing routing policies, the routing policy `return` generates the shortest picker tours on average for order size 5. As already justified, the smaller the order, the better results achieves `return`. For the larger order sizes 15 and 50, the routing policy `largest gap` leads to the shortest picker tours in 24 of 30 settings.
- As highlighted in Table 6, the routing policies behave rather homogenous regarding the different scatter factor settings. Over all order sizes, `largest gap` performs best (aside from the `exact` routing) for all three scatter factor settings. The average tour lengths of the other two routing policies `return` and `traversal` differ only slightly, especially for (1/1/3) and (2/2/2).
- Compared to the results of Petersen and Schmenner (1999) for classical warehouses (without scattered storage), the tour lengths vary over a wider range. In Petersen and Schmenner (1999, Table 6), the largest deviation from optimal routing is 50.5% for *perimeter* and `return`, while for the scatter factor setting (1/1/3) the corresponding value is significantly larger at 113%. Such a greater difference can be observed for most of the routing and storage policy combinations. Consequently, larger cost savings are possible with scat-

tered storage by choosing a suitable combination of the routing and storage policy compared to classical warehouses. However, note that the warehouse layout of Petersen and Schmenner differs from ours. Even though the number of aisles coincides, the warehouse dimension is approximately 2 : 1 in their approach, in contrast to 3 : 4 in ours. Nevertheless, both evaluations find the combination *within-aisle* and `largest gap` the best performing one.

6.3.1 Storage policy perimeter

Given the storage policy *perimeter*, `largest gap` always provides the best results except for one setting that combines (1/1/6) with the order size of 5 (see Table 5). The picker tours generated from the policies `largest gap` and *perimeter* are only about 6% longer than the optimal tours resulting from `exact` and *perimeter*. This is because the rules of `largest gap` are harmonized with the definition of the zones of *perimeter*. Since A-articles are stored around the perimeter and an order consists of 80% A-articles, the procedure of `largest gap` is advantageous where the outer positions of each aisle are cheap to visit. Combined with *perimeter*, the routing policies `return` and `traversal` perform significantly worse than `largest gap`. However, an exception is the already mentioned setting with (1/1/6) and an order size of 5. Here the routing policy `return` leads to better results than `largest gap`. Because only very few articles need to be collected, the typical transition of `largest gap` (where an aisle is entered from both sides and the picker traverses both cross-aisles) enforces an unfavorable selection of pick positions to visit. In contrast, applying the routing policy `return` instructs the picker to traverse only one cross-aisle while entering the necessary aisles to collect the demanded articles. This leads to shorter picker tours for small order sizes. The situation is exactly the opposite with the order size of 50. In this case, many articles from several pick positions need to be collected so that, in the worst case, a typical transition of `return` causes entire aisles to be traversed almost twice, as an aisle must be entered and exited from the same side.

In addition, it can be stated that *perimeter* is particularly suitable for large order sizes. Table 5 shows that, for the order size of 50, *perimeter* is the best choice for half of the settings.

6.3.2 Storage policy across-aisle

The storage policy *across-aisle* performs best in combination with the routing policy `return`, as can be seen from Table 5. Particularly, this affects orders with few C-articles (i.e., small orders) or orders with scattered C-articles where advantageous pick positions are chosen to collect C-articles. For the large order size of 50 and C-articles not being scattered exclusively, it is beneficial to use routing policy `largest gap`. Here, the collection of additional C-articles can easily be appended to the tour. However, for routing policy `return`, the visit of numerous pick positions in zone C induces additional costly path sections, since most aisles must be traversed twice to collect an article of class C. Policy `return` can hardly gain any advantages

from the scattering of the A-articles, since only little distance can be saved by selecting favorable positions of A-articles. As a result, `largest gap` generates shorter tours on average.

In seven of nine settings, the routing policy `traversal` is the worst policy for *across-aisle* because each aisle entered must be traversed completely while many aisles need to be entered to collect all the demanded A-articles. For the order size of 5, storage policy *across-aisle* delivers the best results among all storage policies if combined with `exact` routing.

6.3.3 Storage policy within-aisle

Comparing all 36 combinations of Table 5, *within-aisle* is the storage policy performing best. For nearly 60% of the settings (21 settings), the application of *within-aisle* leads to the shortest picker tours on average. Compared to the other storage policies, *within-aisle* is less sensitive to the rules of the different routing policies. The average tour lengths of *within-aisle* combined with all the routing policies deviate significantly less among each other than the tour lengths produced by any other storage policy in combination with the different routing policies. Nevertheless, `largest gap` is the best routing policy to be combined with *within-aisle*. In seven of nine settings, this combination generates the best results.

For order sizes 15 and 50, the shortest (policy `exact`) picker tours can be obtained with *within-aisle*. In both cases, the best results can be achieved if the C-articles are scattered. The reason is that pick positions for the demanded C-articles can be selected in aisles close to the depot and the picker does not even have to pass through some more distant aisles of the warehouse. For the same reason, *within-aisle* works best, independent of the order size, if the scatter factor setting (1/1/3) is used. Overall, storage policy *within-aisle* leads to the shortest picker tours compared to combinations in which another storage policies is applied.

6.3.4 Storage policy diagonal

The storage policy *diagonal* is a mixture of *across-aisle* and *within-aisle*. Depending on the warehouse layout, it resembles one of the two storage policies more than the other: If the aisles of the warehouse are rather short or the pick positions are arranged narrowly and without spacing, then *diagonal* has strong similarities to *within-aisle*. Otherwise, the warehouse tends to have long aisles and/or large distances between two consecutive aisles so that *diagonal* tends to resemble the storage policy *across-aisle*.

The warehouse layout that we use for the computational study (10 aisles and 40 pick positions per aisle) effectuates that *diagonal* tends to resemble the storage policy *across-aisle*. In seven of the ten aisles, pick positions of zone A can be found. This is also reflected in the results, where the best and worst routing policies for *diagonal* and *across-aisle* are identical except for a single setting (order size of 50 and storage factors (1/1/3)).

7 Conclusions and outlook

This work has focused on the single picker routing problem with scattered storage (SPRP-SS) in combination with standard routing and storage policies. Routing policies restrict possible picker tours to those that obey policy-specific rules, which should preferably allow a fast tour computation and be easy to memorize for the picker. On the theoretical side, we have proven in Sect. 4 that even under these routing policies, the SPRP-SS, i.e., the problem of selecting pick positions in combination with the determination of a shortest picker tour, remains an NP-hard problem.

On the practical side, we have developed an exact solution algorithm for these variants of the SPRP-SS that is both effective and relatively simple to implement. It can be summarized as follows: First, we construct the state space of the DP for the non-scattered SPRP. In Sect. 2, we have derived and detailed these state spaces as modified and restricted versions of the well-known DP of Ratliff and Rosenthal (1983). Second, we introduce into the DP additional aisle actions (in particular for `top`, `bottom`, and `gap`) that become possible in the scattered storage case (see Sect. 3). Third, we set up an IP that models the resulting DP as an origin–destination shortest-path problem. We add demand-covering constraints to ensure that a sufficient number of units of each requested article is collected. Fourth and finally, a standard MIP solver computes an optimal solution to the IP.

The computational evaluation has shown that the main drivers for the practical difficulty of the SPRP-SS are firstly the number of different articles to be picked, secondly the routing policy (the size of the respective state space). The average computation times for pick lists of lengths 5, 15, and 50 are below 49, 228, and 635 milliseconds, respectively, for all routing policies and for a single-block warehouse with ten parallel aisles of 40 cells each. Hence, computation times become irrelevant when the SPRP-SS is considered as a stand-alone problem.

The new and fast exact solution approach has enabled us to precisely analyze and compare different combinations of routing and storage policies. Note that the storage policy, i.e., at which locations in the warehouse articles are stored, determines how a typical instance of the SPRP and SPRP-SS is composed. It does not require a storage policy-specific adaptation of the above solution approach. Therefore, the comparison of all combinations of routing and storage policies focuses on the average tour lengths. The main findings for instances of a standard one-block parallel-aisle warehouse are:

- Neglecting the routing policy `exact`, the combination of the `largest gap` routing policy and the *within-aisle* storage policy lead to the best overall results.
- For class-based storage policies, scattering of C-articles is most advantageous. On the contrary, for uniformly distributed articles, scattering of A-articles performs best.
- In general, if only C-articles are scattered, different routing policies lead to picker tours with remarkable differences in tour length. However, if only A-articles are

scattered, the impact of the different routing policies is substantially less pronounced and leads to only small differences among the tour lengths.

It must be stressed that the presented savings and performance ratios depend on the instance data, which includes the composition of an average pick list as well as the concrete warehouse geometry (number of aisles, number of cells per aisle, distances between aisles and cells and I/O point(s), respectively). Even more, for different warehouse layouts and I/O point configurations (see Table 1), we probably get different results than those obtained in Sect. 6.3 per combination of routing and storage policy. However, we expect similar general findings as the three highlighted above. In this sense, we see our work as a fundamental step to introduce a methodology that is widely applicable to perform experiments and quantify savings in average picker tour lengths for a particular warehousing setting.

Accordingly, we see several avenues for interesting further research: Alternative warehouse settings other than the single-block parallel-aisle warehouse with a unique I/O point should be analyzed. Since DP algorithms are known for most of them (see Table 1), a solution approach adapted from the one presented here should also be tested and similar combinations of routing and storage policies should be analyzed. One complication is that, to our knowledge, class-based storage policies have not yet been defined for two-block, multi-block, and other warehouse layouts. Even for warehouses where the I/O point is not in one of the corners and for warehouses with multiple drop-off points, the definition of the storage policies is unclear. This necessary specification should be done first. In addition, the storage policies could be refined and alternative storage policies could be empirically analyzed using the methodology proposed in this paper. For example, articles that are ordered together more often than other articles could be assigned to pick positions that are closer together. For class-based storage policies, one may vary the scatter factors within class C. An analysis of how article-specific scatter factors should be chosen is worth further investigation when class-based storage policies are replaced by full-turnover storage. Finally, if SPRP-SS appears as a subproblem, e.g., in the joint order batching and picker routing problem (Wahlen and Gschwind 2023), the presented IP-based solution algorithm might become too slow. Therefore, finding an even more effective solution algorithm for the SPRP-SS remains an algorithmic challenge.

Funding Open Access funding enabled and organized by Projekt DEAL.

Data availability All instances are online available at https://logistik.bwl.uni-mainz.de/reserach/benchmarks/.

References

Behnisch P, Glock CH, Grosse EH, et al (2017) Auf dem Weg zum Warehouse 4.0?: Zum aktuellen Stand der Automatisierung in der Lagerhaltung. Technical Report 84808, Institue for Business Studies (BWL), Department of Business Administration, Economics and Law, Darmstadt Technical University, (**in German**)

Bock S, Boysen N (2023) Routing replenishment workers: the prize collecting traveling salesman problem in scattered storage warehouses. INFORMS J Comput 36(1):3–20

Boysen N, de Koster R, Weidinger F (2019) Warehousing in the e-commerce era: a survey. Eur J Oper Res 277(2):396–411. https://doi.org/10.1016/j.ejor.2018.08.023

Çelk M, Süral H (2014) Order picking under random and turnover-based storage policies in fishbone aisle warehouses. IIE Trans 46(3):283–300. https://doi.org/10.1080/0740817x.2013.768871

Cormier G, Gunn EA (1992) A review of warehouse models. Eur J Oper Res 58(1):3–13. https://doi.org/10.1016/0377-2217(92)90231-W

Cornuéjols G, Fonlupt J, Naddef D (1985) The traveling salesman problem on a graph and some related integer polyhedra. Math Program 33(1):1–27. https://doi.org/10.1007/BF01582008

de Koster R, van der Poort E (1998) Routing orderpickers in a warehouse: a comparison between optimal and heuristic solutions. IIE Trans 30(5):469–480. https://doi.org/10.1023/a:1007599307171

de Koster R, Le-Duc T, Roodbergen KJ (2007) Design and control of warehouse order picking: a literature review. Eur J Oper Res 182(2):481–501. https://doi.org/10.1016/j.ejor.2006.07.009

Daniels RL, Rummel JL, Schantz R (1998) A model for warehouse order picking. Eur J Oper Res 105(1):1–17. https://doi.org/10.1016/S0377-2217(97)00043-X

Garey MR, Johnson DS (1979) Computers and intractability: a guide to the theory of NP-completeness. Freeman, San Francisco

Gibson DR, Sharp GP (1992) Order batching procedures. Eur J Oper Res 58(1):57–67. https://doi.org/10.1016/0377-2217(92)90235-2

Goeke D, Schneider M (2021) Modeling single-picker routing problems in classical and modern warehouses. INFORMS J Comput 33(2):419–835. https://doi.org/10.1287/ijoc.2020.1040

Gu J, Goetschalckx M, McGinnis LF (2007) Research on warehouse operation: a comprehensive review. Eur J Oper Res 177(1):1–21. https://doi.org/10.1016/j.ejor.2006.02.025

Gue KR, Meller RD (2009) Aisle configurations for unit-load warehouses. IIE Trans 41(3):171–182. https://doi.org/10.1080/07408170802112726

Gutin G, Punnen A (eds) (2002) The traveling salesman problem and its variations, combinatorial optimization, vol 12. Kluwer, Dordrecht

Hall RW (1993) Distance approximations for routing manual pickers in a warehouse. IIE Trans 25(4):76–87. https://doi.org/10.1080/07408179308964306

Hausman WH, Schwarz LB, Graves SC (1976) Optimal storage assignment in automatic warehouse systems. Manag Sci 22:629–638. https://doi.org/10.1287/mnsc.22.6.629

Henn S, Koch S, Gerking H et al (2013) A U-shaped layout for manual order-picking systems. Logist Res 6(4):245–261. https://doi.org/10.1007/s12159-013-0104-6

Heskett JL (1963) Cube-per-order index-a key to warehouse stock location. Transp Distrib Manag 3(1):27–31

Heßler K, Irnich S (2022a) Modeling and exact solution of picker routing and order batching problems. Technical report LM-2022-03, chair of logistics management, Gutenberg School of Management and Economics, Johannes Gutenberg University Mainz, Mainz, Germany, https://logistik.bwl.uni-mainz.de/files/2022/04/LM-2022-03.pdf

Heßler K, Irnich S (2022b) A note on the linearity of Ratliff and Rosenthal's algorithm for optimal picker routing. Oper Res Lett 50(2):155–159. https://doi.org/10.1016/j.orl.2022.01.014

Heßler K, Irnich S (2024) Exact solution of the single picker routing problem with scattered storage. INFORMS J Comput. https://doi.org/10.1287/ijoc.2023.0075

Khan I, Maurer O, Pätzold J, et al (2024) Joint order selection, allocation, batching and picking for large scale warehouses. Technical report, arXiv:2401.04563, https://doi.org/10.48550/arXiv.2401.04563

Löffler M, Boysen N, Schneider M (2022) Picker routing in AGV-assisted order picking systems. INFORMS J Comput 34(1):440–462. https://doi.org/10.1287/ijoc.2021.1060

Löffler M, Boysen N, Schneider M (2022) Picker routing in AGV-assisted order picking systems. INFORMS J Comput 34(1):440–462. https://doi.org/10.1287/ijoc.2021.1060

Masae M, Glock CH, Vichitkunakorn P (2019) Optimal order picker routing in the chevron warehouse. IISE Trans 52(6):665–687. https://doi.org/10.1080/24725854.2019.1660833
Masae M, Glock CH, Grosse EH (2020) Order picker routing in warehouses: a systematic literature review. Int J Prod Econ 224:107564. https://doi.org/10.1016/j.ijpe.2019.107564
Masae M, Glock CH, Vichitkunakorn P (2020) Optimal order picker routing in a conventional warehouse with two blocks and arbitrary starting and ending points of a tour. Int J Prod Res 58(17):5337–5358. https://doi.org/10.1080/00207543.2020.1724342
Masae M, Glock CH, Vichitkunakorn P (2021) A method for efficiently routing order pickers in the leaf warehouse. Int J Prod Econ 234:108069. https://doi.org/10.1016/j.ijpe.2021.108069
Miller CE, Tucker AW, Zemlin RA (1960) Integer programming formulations and traveling salesman problems. J Assoc Comput Mach 7:326–329. https://doi.org/10.1145/321043.321046
Öztürkoğlu Ömer, Gue KR, Meller RD (2012) Optimal unit-load warehouse designs for single-command operations. IIE Trans 44(6):459–475. https://doi.org/10.1080/0740817x.2011.636793
Öztürkoğlu O, Hoser D (2019) A discrete cross aisle design model for order-picking warehouses. Eur J Oper Res 275(2):411–430. https://doi.org/10.1016/j.ejor.2018.11.037
Pansart L, Catusse N, Cambazard H (2018) Exact algorithms for the order picking problem. Comput Operat Res 100:117–127. https://doi.org/10.1016/j.cor.2018.07.002
Park YH, Webster DB (1989) Design of class-based storage racks for minimizing travel time in a three-dimensional storage system. Int J Prod Res 27(9):1589–1601. https://doi.org/10.1080/00207548908942641
Petersen CG (1995) Routeing and storage policy interaction in order picking operations. In: Decision sciences institute proceedings, pp 1614–1616
Petersen CG (1997) An evaluation of order picking routeing policies. Int J Oper Prod Manag 17(11):1098–1111. https://doi.org/10.1108/01443579710177860
Petersen CG, Schmenner RW (1999) An evaluation of routing and volume-based storage policies in an order picking operation. Dec Sci 30(2):481–501. https://doi.org/10.1111/j.1540-5915.1999.tb01619.x
Petersen CG, Aase GR, Heiser DR (2004) Improving order-picking performance through the implementation of class-based storage. Int J Phys Distrib Logist Manag 34(7):534–544. https://doi.org/10.1108/09600030410552230
Ratliff HD, Rosenthal AS (1983) Order-picking in a rectangular warehouse: a solvable case of the traveling salesman problem. Operat Res 31(3):507–521. https://doi.org/10.1287/opre.31.3.507
Roodbergen KJ (2001) Layout and routing methods for warehouses. PhD thesis, RSM Erasmus University, Rotterdam, The Netherlands
Roodbergen KJ, de Koster R (2001) Routing methods for warehouses with multiple cross aisles. Int J Prod Res 39(9):1865–1883. https://doi.org/10.1080/00207540110028128
Roodbergen KJ, de Koster R (2001) Routing order pickers in a warehouse with a middle aisle. Eur J Oper Res 133(1):32–43. https://doi.org/10.1016/s0377-2217(00)00177-6
Scholz A, Wäscher G (2017) Order batching and picker routing in manual order picking systems: the benefits of integrated routing. Central Eur J Oper Res 25(2):491–520. https://doi.org/10.1007/s10100-017-0467-x
Scholz A, Wäscher G (2017) Order batching and picker routing in manual order picking systems: the benefits of integrated routing. Central Eur J Oper Res 25(2):491–520. https://doi.org/10.1007/s10100-017-0467-x
Su Y, Zhu X, Yuan J et al (2023) An extensible multi-block layout warehouse routing optimization model. Eur J Oper Res 305(1):222–239. https://doi.org/10.1016/j.ejor.2022.05.045
van Gils T, Caris A, Ramaekers K et al (2019) Formulating and solving the integrated batching, routing, and picker scheduling problem in a real-life spare parts warehouse. Eur J Oper Res 277(3):814–830. https://doi.org/10.1016/j.ejor.2019.03.012
Wahlen J, Gschwind T (2023) Branch-price-and-cut-based solution of order batching problems. Transp Sci 57(3):756–777. https://doi.org/10.1287/trsc.2023.1198
Weidinger F (2018) Picker routing in rectangular mixed shelves warehouses. Comput Oper Res 95:139–150. https://doi.org/10.1016/j.cor.2018.03.012
Weidinger F, Boysen N (2018) Scattered storage: how to distribute stock keeping units all around a mixed-shelves warehouse. Transp Sci 52(6):1412–1427. https://doi.org/10.1287/trsc.2017.0779
Weidinger F, Boysen N, Schneider M (2019) Picker routing in the mixed-shelves warehouses of e-commerce retailers. Eur J Oper Res 274(2):501–515. https://doi.org/10.1016/j.ejor.2018.10.021

Authors and Affiliations

Laura Lüke[1] · Katrin Heßler[2] · Stefan Irnich[1]

✉ Laura Lüke
lueke@uni-mainz.de

Katrin Heßler
katrin.hessler@dbschenker.com

Stefan Irnich
irnich@uni-mainz.de

[1] Chair of Logistics Management, Gutenberg School of Management and Economics, Johannes Gutenberg University Mainz, Jakob-Welder-Weg 9, 55128 Mainz, Rhineland-Palatinate, Germany

[2] Global Data & AI, Schenker AG, Kruppstr. 4, 45128 Essen, North Rhine-Westphalia, Germany